Automatic Morphosyntactic Annotator System for Kannada

A thesis to be submitted for the degree of

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE

By

BHUVANESHWARI C. MELINAMATH

(07MCPC03)

School of Computer and Information Sciences

University of Hyderabad

Hyderabad

2016

CERTIFICATE

This is to certify that Bhuvaneshwari **C. Melinamath** bearing **Reg.No. 07MCPC03** has carried out her research work in School of Computer and Information Sciences, University of Hyderabad for the thesis entitled **"Automatic Morphosyntactic Annotator System for Kannada",** in partial fulfilment of the requirements for the award of Doctor of Philosophy and she has successfully completed all the pre-requisites for thesis submission.

The thesis is also free from any plagiarism.

The School recommends this thesis for further evaluation by external examiners as per the rules of the University.

Prof H. Mohanty
Coordinator
School of Computer and Information Sciences
University of Hyderabad.

Prof. Arun Agarwal
Dean
School of Computer and Information Sciences
University of Hyderabad

DECLARATION

I, **Bhuvaneshwari C. Melinamath,** bearing Reg. No. **07MCPC03**, hereby declare that this thesis entitled **"Automatic Morphosyntactic Annotator System for Kannada,"** is carried out by me at School of Computer and Information Sciences, University of Hyderabad. It is a bonafide research work and is free from any plagiarism. I also declare that it has not been submitted previously in part or in full to this University or any other University or Institution for the award of any degree or diploma. I hereby agree that my thesis can be submitted in Shodganga/INFLIBNET.

A report on plagiarism statistics from the Librarian, University of Hyderabad, is enclosed.

Date:

Name: Bhuvaneshwari C. Melinamath
Regd. No: 07MCPC03

Dedicated to my Beloved Husband and My Mother

-whose unparalled sacrifice is the motivating force for me

ACKNOWLEDGEMENT

With blessings of Shree Gurupadeshwara and Shree Satya Sai baba, I am in position to acknowledge the people who are with me through this journey of Research and I am certainly the recipient of the dedication, devotion, insight and hard work of the people around me. My heartfelt Gratitude to all those for what I have received over this period.

First of all, I would like to express my deepest sense of gratitude and thanks to my coordinator Prof. Hrushikesha Mohanthy. It is a great honour for me to have an eminent and versatile personality as my coordinator.

I would like to express my sincere gratitude to my Dean & Head of the Department Prof. Arun Agarwal for his exceptional patience, tender-hearted support and smiling encouragement all through. I took a lot from his scrupulous knowledge and meticulous experience during my research work. His trust in my abilities helped me to overcome many difficulties.

My sincerest thanks to my Doctoral Committee members Prof. Udgata and Dr. Durga Bhavani. They were always encouraging and helpful in my research. My sincerest thanks to Dr. Durga Bhavani madam for many insightful conversations and support and encouragement throughout my Ph.D.

I would like to express my sincere gratitude to the Vice Chancellor, Deputy Registrar and all officials of HCU, Hyderabad, for their cooperation and encouragement.

I sincerely thank Dr. M. B Patil, President B.L.D.E.A's Association, and honourable irrigation minister of Karnataka state for providing me an opportunity to pursue Ph.D. It is a great honour for me to work under such an eminent and versatile personality. Thanks to his generous personality. I am also thankful to Smt. Asha. M . Patil.

I sincerely thank my beloved former principal, of B.L.D.E.A's engineering college, Dr. Katkol whose multidimensional support and unparalleled cooperation kept me active throughout. Who is the true source of inspiration for the many researchers of our generation.

I would like to express my sincere gratitude to Dr.V.P.Huggi Principal BLDEA’s Engg. College Vijayapur, for his kind co-operation whenever required.

My sincerest thanks to Prof. C. Raghavendra Rao and Prof. Rukma Rekha , for their valuable suggestions during my grievance committee meetings.

My sincerest thanks to my well wisher and my beloved principal Dr. Basavaraj Anami, Principal KLE Engineering College for his valuable support and cooperation and encouragement during my research work. It would have been impossible without the intellectual support of Dr. Anami.

I am thankful Shri G.N. Deshapande Principal V B Darbar College Bijapur, Shri Vasudev Herkal district reporter Samyukta Karnatak news paper Bijapur and Shri Manjunath Konasur district reporter Vijayakarnatak News paper.

My sincere thanks to Dr. G B Sajjan for making grammatical corrections and for giving valuable suggestions.

I am indebted to Mr. Nandi Chinnikumar student president who supported me a lot towards my completion of my thesis. I also thank fellow researchers Mr. Priyadarshi, Mr. Abdul, Prof. S M. Hatture and Prof. Vilas Naik who shared their valuable time.

My special thanks goes to my dearest husband Mr. Allamprabhu and my Mother Smt. Shantabai who helped me whenever I was in great need. Without them my work is incomplete. I am forever indebted to my parents and family members for their understanding, endless patience and encouragement when it was most required.

BHUVANESHWARI C MELINAMATH

Chapter 1

Introduction

1.1 Natural Language Processing

Natural Language Processing (NLP) is an area which is concerned with the processing of human language from the computational perspective. The complexity involved in a natural language is more, as it involves cognition, psychology philosophy etc. Hence processing of natural languages is difficult. The goal of NLP is to analyze, understand the natural languages used by humans and to encode linguistic knowledge into rules or other forms of representation. Natural language texts in electronic form have become the principal medium for communication. Since electronic documents and texts are natural language constructs.

The main difficulty in natural language processing tasks is perhaps its ambiguity. Ambiguity in natural language pervades virtually all aspects of language analysis. Sentence analysis in particular exhibits a large number of ambiguities that demand adequate resolution before the sentence can be understood. Most of the language processing applications like Information Retrieval (IR), Information Extraction (IE), Question-Answering system, Speech Recognition and Synthesis, Text Summarization and Machine Translation (MT) are affected by the highly ambiguous nature of natural language. Recently, statistical and machine learning algorithms have taken a lead over complex linguistic grammar. There has been great progress in natural language processing through the use of statistical methods trained on a large mass of tagged corpora. Statistical tagging approaches are able to tackle the ambiguity problem by assigning a probability to each word. One more drawback of statistical algorithms is they require large quantity of annotated data for training.

Reduplicative, lexical doublets and echo compounds are characteristic features of our languages rather than, say English, which is why a comparative study of English and Indian languages in this area is of great value. A comparative study would yield rich information regarding the semantic structure of languages, apart from being useful for translation systems, etc.

Automatic annotation task is central to much research related to the field of corpus linguistics. The significance of large annotated corpora in present day NLP is widely required. Annotated corpora serve as an important tool for investigators of natural language processing, speech recognition and other related areas. It proves to be the basic building block for constructing statistical models for automatic processing of natural languages. The process of annotation can be performed at different levels like word level, phrase level, clause level etc. *To perform annotation, various resources like tagset, dictionaries, high performance morphological analyzers, taggers and NER systems are required. Lack of these resources for Kannada drove us to move in this direction.* ***Designing any NLP tool for Kannada like languages is quite challenging as Kannada language has very complex morphological process.***

1.2 About Kannada Language

Kannada is a language of Dravidian family spoken mainly in the southern part of India and ranks third among Indian languages in terms of number of speakers. **Kannada is a highly inflectional and agglutinating language providing one of the richest and challenging set of linguistic and statistical features.** Kannada is a language characterized by a rich system of inflectional morphology and a productive system of derivation, SaMdhi and compounding. Although Indian languages are characterized by an ancient and rich literature and are spoken by large numbers of people, progress in Indian languages in terms of technology has been very slow. It is only recently that significant quantitative and computational work on Indian languages has started taking off the ground. Kannada language has an ancient history of more than 2,000 years. **The Dravidian languages form a separate group from the Indo-Aryan languages like Hindi.**

Kannada language uses 49 phonemic letters, divided into 3 groups: swaragaLu "vowels" 14 in number as shown in table 1.2, vyaNjangaLu " consonants" 34 in number as shown in table 1.1 and yogavaahakagaLu (neither consonant nor vowel) are 2 in number anusvara “aM " and visarga “ah”.

Table 1. 1. Kannada Consonants with Transliteration

ಕ ಖ ಗ ಘ ಙ	ತ ಥ ದ ಧ ನ
ka kha ga gha n`a	ta tha da dha na
ಚ ಛ ಜ ಝ ಞ	ಪ ಫ ಬ ಭ ಮ ಯ ರ
ca cha ja jha N`a	pa pha ba bha ma ya ra
ಟ ಠ ಡ ಢ ಣ	ಲ ಳ ವ ಷ ಶ ಸ ಹ
Ta Tha Da Dha Na	la La va s`a sha sa ha

Table 1.2. Kannada Vowels with Transliteration

ಅ ಆ ಇ ಈ ಉ ಊ ಋ ೠ ಎ ಏ ಐ ಒ ಓ ಔ
a aa i i i u uu R R e ee ai o oo au
Anusvaara（Binduor Sonne）ಅ◌ಂ aM Visarga ಅ ◌ಃ ah.

1.2.1 Complexity of Kannada Language

Kannada, like other Dravidian languages has complex morphology and a single root can lead to the formation of a very large number of surface word forms. Words in Dravidian languages in general, are an order of magnitude more complex than those in Indo-Aryan languages. Kannada words tend to be long and complex. The mean word length in terms of bytes (or ’characters’) is 11.29 and the standard deviation is 4.41 as predicted in (Akshara Bharati, Sangal and Sushma M Bendre,1998). English words have a mean length of 8.18 with a standard deviation of 3.12. The calculations for English are based on a 3 Million words English Corpus derived from the British National Corpus (BNC) by random selection.

1.2.2 Richness in Morphology

The main reason for richness in the morphology of Kannada and other Dravidian languages is, that a significant part of grammar that is handled by syntax in English is handled within morphology in Kannada. Phrases including several words in English would be mapped on to a single word in Kannada. Thus the Kannada word forms like ಬಂದನು‘baMdanu’ (he came), ಬರುವನಾ **‘baruvanaa’** (will he come?), **ಬಂದರೆ‘baMdare’** (if (he /she/it/they/I/w/eyou) come), **ಬರಬಲ್ಲನು ‘baraballanu’** (he is able to come), **ಬರುವದಿಲ್ಲವೇನು? ‘baravadillavenu’** (will he/she/it/you/they will not come?) are representing single word with different semantics and are written and spoken as atomic units. A single verbal root in Kannada can lead to formation of a few thousands of word forms. *Complicated word formation using e*xternal *saMdhi* (that is, conflation between two or more complete word forms), leads to very large number of types and the *type-token ratio may be expected to be very high too.* These words are made up of several morphemes conjoined through complex morphophonemic processes. *Kannada in particular, and Dravidian languages in general, are among the most complex languages in the world at the level of morphology, perhaps comparable only to Finnish and Turkish, which are considered as most complex morphology languages of the world.*

1.2.3 Free Word Order

Kannada is a free word order language. Word order refers to definite positional constraints for the various constituents in a sentence, and more importantly these positions carry syntactic information. *English specifies a fixed word order.* Linear order of words and phrases in a sentence is so significant that changing the word order may render the sentence ungrammatical or anomalous. An important characterization of any language is the relative freeness of the word order that it exhibits. English is a SVO (subject verb object) language, while Indian languages are considered to be SOV languages but with a relatively free word-order. *Since Kannada is a free word order language, that is, the meaning of the sentence is not going to change by the changing of the place of its constituents.* For example, all of the following Kannada sentences shown below mean essentially the same thing.

Subject1 Object Subject2 Verb

↓ ↓ ↓ ↓

ರಾಮನು ಶಾಲೆಗೆ ಸೀತೆಯೊಡನೆ ಹೋದನು ...(1)

raamanu shaalege siiteyoDane hoodanu (Transliteration)

Rama to school with Sita went (English Mapping)

Rama went to school with Sita (English)

ಸೀತೆಯೊಡನೆ ರಾಮನು ಶಾಲೆಗೆ ಹೋದನು ...(2)

siiteyoDane raamanu shaalege hoodanu

ಶಾಲೆಗೆ ಹೋದನು ರಾಮನು ಸೀತೆಯೊಡನೆ ...(3)

shaalege hoodanu raamanu siiteyoDane

ರಾಮನು ಹೋದನು ಶಾಲೆಗೆ ಸೀತೆಯೊಡನೆ ...(4)

raamanu hoodanu shaalege siiteyoDane

In English the transposition of words in the sentences produces ungrammatical sentences, whereas in Kannada all 3 sentences gives correct meaning, its support for free word order and increases the grammar complexity. In the next section various approaches used in the annotation process are discussed.

1.3 Existing Approaches for Annotation

Annotation is a process of adding grammatical information like noun, verb etc., to the words in the text. The tagging process may be supervised or unsupervised. Both processes follow the grammar of the language, complexity of handling grammar increases as we move from word level to sentence level. Unsupervised tagging by (Goldsmith and Gaussier, 1999) and (Creutz M, 2003) states that developing grammar formalism for word grammar or grammar at higher level is extremely laborious. The generally used approaches for Annotation are as follows.

i) Rule based approach for tagging

ii) Machine learning based approach for tagging

iii) Hybrid model based approach for tagging

1.3.1 Rule Based Tagging

Rule based tagging is a linguistic approach, in which two stages are involved. In the first stage the dictionary is used to assign potential part of speech for each word. In the second stage a *large list of hand crafted rules are used to assign part of speech for each word.* Sample Rules used in English are shown in below example 1 and 2.

Example 1: ART-V rule: A tag ART (article) cannot be followed by a tag V (verb).

The book.

The: {ART}

Book: {N, V} − > {N}, here book is a word with two tags n-noun and v- Verb, this ambiguity of assigning tag can be resolved by ART-V rule specified above, which says article cannot be followed by verb, hence book is assigned the tag noun.

Example 2: N-IP rule:

A tag N (noun) cannot be followed by the tag IP (interrogative pronoun)

. ...man who...

Man: {N}

Who: {RP, IP} − > {RP}

1.3.1.1 Rule Based POS Taggers for Indian Languages

Rule Based Part of Speech Tagger for Hindi developed by (Vishal Goyal et al.) is evaluated over a corpus of 26,149 words with 30 different standard part of speech tags for Hindi. The evaluation of the system is done on different domains of the Hindi Corpus. These domains include news, essay, and short stories. The system achieved an accuracy rate of 87.55%.A Rule based POS tagger for Telugu was developed by (CALTS UoH, 2008). The tagset used by them for POS Tagging consists of 53 tags. They formulated 524 rules for POS disambiguation. A Rule based POS tagger for Tamil was developed by AU-KBC research centre. The tagset used by them consists of 26 tags. They used 97 rules for POS disambiguation. It has given a precision of 92%.

1.3.2 Stochastic Tagging

Stochastic tagging techniques make use of corpus. Stochastic tagging techniques can be of two types depending upon the training data. They are supervised or unsupervised. The most common stochastic tagging technique uses a Hidden Markov Model (HMM). HMM model parameters are calculated from the tagged training corpus. POS tags are represented as states of model. The states are hidden and state sequence for a given word sequence is estimated usingViterbi algorithm, which is most likely to generate the observed word sequence. Stochastic unsupervised POS tagging technique requires no tagged training corpus, but instead uses sophisticated computational methods to automatically induce tagsets and based on these they calculate the probabilistic values needed by the stochastic tagger.

1.3.2.1 Stochastic Based POS Taggers for Indian Languages

Stochastic tagging techniques make use of training corpus. Stochastic tagging techniques can be either supervised, unsupervised or hybrid. HMM based POS tagger are implemented for Indian languages by different researchers like (Asif Ikbal, and Shivaji Bandyopadhyay, 2007), (Sandipan Dandapat and Sudeshna Sarkar, 2006) for Bengali and (Aniket Dalal, Kumar Nagaraj, Uma Sawant, and Sandeep Shelke, 2006) developed mainly for Hindi. Here they make use of pre-tagged training corpus. Handling of unknown words, words which are not present in the corpus is performed based on suffixes. SPSAL workshop conducted in 2007 by IIIT-Hyderabad had an accuracy of POS tagging using machine learning approach for Hindi, Bengali and Telugu and reported an accuacy of 73.93%, 72.35% and 71 % respectively. The accuracy for Telugu was low as compared to that of Hindi and Bengali.

1.3.3 Hybrid Model for Tagging

One way to achieve hybrid model tagging is to have combination of both supervised and unsupervised methods. Other ways of hybrid model is to have a combination of supervised and rule based method. Hybrid POS tagger was developed for Tamil by (Vijay and Shobha, 2012). The test data used by them consists of 6098 words out of which only 3547 are correctly tagged. Hybrid POS tagger was developed for Bengali by (Shivaji Bandopadhay, 2006). Their approach is a combination of both supervised and unsupervised stochastic techniques for training HMM.

1.4 Motivation

In English, lot of work related to tagging has been accomplished. But Indian languages are still lagging behind in annotation/tagging and also in other NLP related works, like machine translation systems, POS taggers, chunkers. Status of Kannada language is also the same. Till date, *Kannada does not have any computational grammar, which defines a precise algorithmic way of saying whether a given word is a valid word or not*. Many grammar books written for Kannada explain the pedagogical grammar, but none of them gives an algorithmic way of expressing these rules which can test the valid Kannada words, so that one can try to implement these rules on the computer.

Annotation is still considered to be a budding topic of research in the field of NLP as considered with respect to Kannada and much of the work needed is in this regard. Kannada is a highly inflected natural language. *Kannada is a resource poor language in terms of NLP resources. Annotated Corpora, name dictionaries, good morphological analyzer, etc. are not yet available in Kannada language.* Few existing NLP systems for Kannada like (Shambhavi, 2011), (K.N.Murthy, 1999), (Uma Maheshwara and G Parameshwari, 2010) and IIIT-Hyderabad are not sufficient. The following observations mentioned below motivated us to move in this direction.

- Lack of exhaustive morphological analyzer/generator handling, derivational morphology as well as inflectional morphology. The existing morphological analyzer/generator systems use flat tagset.
- Use of hierarchical tag set in morph module is missing in the existing systems. Lack of good hierarchical Part of Speech (POS) tag set.
- Lack of named entity recognizer (NER) system for Kannada motivated us to move in the direction of developing these lexical resources for Kannada.
- Lack of good Annotation systems. Annotation is an essential and challenging task in Natural Language Processing (NLP). The motivation behind developing an exhaustive annotator system is to exploit our language computationally and to provide means for developing a tagged corpus.
- Annotated corpora serve as an important tool for investigators of natural language processing, speech recognition and other related areas. Annotated corpora prove to be the basic building block for constructing statistical models for automatic processing of natural languages.

This situation motivated us towards the development of automatic morphosyntactic annotator system for Kannada language. Some of the applications of the Annotated text include Information Extraction (IE), Machine Translation. The main motivation of this work is to achieve a functionally automated annotation system for Kannada language.

1.5 Problem Statement

The objective of our work isto create a new computational model within the framework of finite state technology that will account for word formation processes in Kannada language. The work carried out aims at development of methodologies for classification of words in to their part of speech categories by implementing “Automatedmorphsyntactic annotator system (AMAS) for Kannada”. However, in the process of fulfilling this aim, a number of subsidiary aims led to the development of other important lexical resources for Kannada language like design and development of hierarchical tag set, developing an electronic dictionary, developing an exhaustive morphological analyzer/generator and developing hybrid named entity recognizer. NER is developed by applying judicious combination of linguistic and statistical techniques like HMM and CRF.

1.6 Scope and Objectives Fulfilled

Description of our language within the framework of computational linguistics is itself a legitimate research objective, because it meaningfully occupies the space between descriptions offered by grammars of the traditional type and those with restrictive theoretical commitments. The work carried out has its scope limited to handling of foreign words and limited compound words. The main aim of this work is to achieve functional automated annotation system for Kannada.The gist of the work carried out is briefed below.

i. Developed Hierarchical POS tagset for Kannada language. Each word is tagged with all its possible features. Required NLP applications can make use of the extent of information required for their needs. MT applications require more features. *It is better to develop the details in early stages rather than at later stages when required.*

ii. Developed semiautomatic methodology for building an electronic dictionary. A dictionary of 30000 words using hierarchical POS tagset is developed for the first time. Use of hierarchical tagset is more advantageous than flat tagset in applications like morphology, parsing and Machine Translation. The dictionary lists the exceptions and words with irregular morphology. The features of a word like countable/uncountable and *real “u”* are stored in the dictionary. These features control the morphological generator to be over general.

iii. Developed a methodology for computational grammar using finite state technology, by implementing morphological analyzer/generator, which defines a precise algorithmic way of saying, whether a given word is a valid word. A detailed analysis of word formation is given as morpheme by morpheme; hence morph module can be used as spell checker and also as language learning tool.

iv. Developed Rule based methodology for Named Entity Recognizer (NER).

v. Developed machine learning methodology for NER using Hidden Markov Model (HMM).Another statistical approach, Conditional Random Field (CRF) which performs better than HMM, is also experimented with NER.

1.7 Outline of Methodology

The main aim of this thesis is to achieve a functional Automated Annotator System (AMAS) for Kannada. The AMAS architecture is shown in figure 5.1. The methodology adopts a hierarchical tagset module. This is the first system built using a hierarchical *tagset for the Kannada language.* The general methodology adopted in the work consists of five stages, preprocessing involves transliteration of the Unicode or ISCII to Roman and remove unwanted blank lines or characters. Next step is tokenization of the sentence**.** Then searching out the words in the dictionary, if the word is found in the dictionary then the dictionary tag is assigned, otherwise it is passed to the morphological analyzer/generator module for analysis of possible inflections. If the word is not analyzed then it is passed to NER Module to assign a tag for proper noun.

The system takes a Kannada sentence as input and gives POS tag/tags for each word in the sentence. Morphological analyzer is used for tagging the inflected words and uninflected words are tagged using the dictionary. NER module tags proper nouns in the sentences. POS tag indicates grammatical class/category of a word such as noun, verb, etc. Here a hierarchical tag is used for the first time instead of the flat tag. The few implementations that pre-exists, like (Shambhavi, 2011), (K N Murthy, 1999), (Shalini et al., 2007), IIIT-Hyderabad have used flat tagset. These systems uncover many of the morphological aspects like handling of clitics, derivational morphology, verb formation process using aspect auxiliary, use of vocatives, modal auxiliaries.

1.7.1 Implementation Morphological Analyzer/generator

"Morphological analyzer/generator developed here is the first system developed using a hierarchical POS tagset. This is the first experiment among Indian languages and also in Kannada". FST uses non deterministic finite automata (NFA). The goal is to accept all valid word forms in a given sentence. The states and transitions are listed in table form. Our table driven implementation consists of syntactic State, next state, set of syntactic productions. The table of syntactic productions consists of a) A category symbol, b) A condition state symbol and c) A result state symbol. Results are encouraging with respect to noun as compared to verbs. More than 90% accuracy is obtained. Apart from inflection morphology, derivational morphology, clitics are also handled here. We have developed bidirectional morphological analyzer/generator on single finite state transducer. It is observed that there are around 250 different word forms for a single noun in Kannada as compared to English, wherein only 2 forms exist, similarly there are more than 3000 word forms exists for single verb in Kannada. It is a system representing detailed morph syntactic and semantic information in such a way that it is computationally useful. The main function of this system is to identify and enumerate all the construction types (within the linguistic limits) of a particular language, down to all degrees of detail. The output of the morph is represented in such a manner that they could be effectively compared cross-linguistically. Thus in this system, the construction types are represented by strings of letters and hyphens composed of labels. Each construction is displayed from the top to bottom level shows its hierarchy, in Parts of Speech, etc. This approach of construction labeling would be helpful in developing applications where detailed information is required like machine translations.

1.7.2 Development of Hierarchical Part of Speech (HPOS) TagSet

We are proposing a Hierarchical Part of Speech (HPOS) tagset for Kannada. We present here a hierarchical tagset based on syntax. There is no workable POS

(Part of Speech) tagset or tagger for Kannada till today. POS tagger serves as the fundamental building block for NLP researches. We follow ***EAGLES (Expert Advisory Group on Language Engineering Standards)*** as guideline with modifications as required for our Kannada Language. Designing the tagset for Kannada is challenging due to its inherent complex word formation process in Kannada. We are handling a contingent feature *which is a special feature for Kannada, voice modifiers, causative feature, reflexive features, clitics information, and semantic information like countable/uncountable features all these are considered in our tagset design.* Important features like usage of aspect auxiliaries in the formation of complex verb derivations are handled in our tagset. Use of modal auxiliaries is a very important aspect in generation of complex verb forms. This feature is also captured in our tag set, marking of transitivity, intransitivity are also handled at dictionary level. Fine grained classification of adjectives and adverbs is useful information in grouping chunk elements, which adjective follows which adjective is important information in chunking application. For example basic adjectives are not followed by ordinal adjectives. This is useful information while forming chunk groups. Hence adjectives should be classified into subclasses.

1.7.3 Developing an Electronic Dictionary Using HPOS tag

Developing an electronic dictionary, using hierarchical tag is a valuable effort because each word has to be marked for its gender, number, countable and uncountable features. *Till today an electronic dictionary using the hierarchical tagset is not available for Kannada. Hierarchical approach is a new attempt in this direction*. The available dictionary at IIIT-Hyderabad website is bilingual (Kannada-Hindi) and developed with a flat tagset. Treatment of categories considered varies widely with respect to our principles, and there are many inconsistencies. It was not directly usable. Another dictionary by Prof K. N. Murthy was a monolingual dictionary but phrases were treated as words and developed with a flat tagset. The size of the dictionary was 10000 words. Same is case with a few online Kannada dictionaries. It is shown here that the available resources are inadequate and full of errors. Simple techniques such as heuristic pattern matching, regular expressions, frequency vs. rank kind of knowledge, stemming and filtering technique are used in building this dictionary. An electronic dictionary of 30000 plus words is built. A total of 126 different cases pratyayas, information regarding word ending patterns are used in developing this dictionary. Electronic dictionaries are assets for computational linguistics. Semiautomatic methods are used for building dictionaries from electronic text corpora. We have used Department of Electronics (DoE), Central Institute of Indian Languages (CIIL) corpus for this experimentation. We have used a hierarchical tag for tagging each word in the dictionary instead of relying on a flat tagset.

1.7.4 Implementation Named Entity Recognizer Using Rule Based Approach

Named entity recognizer using rule based approach is implemented using handcrafted rules, rule based approach requires knowledge of native language to design the rules and requires more time than machine learning approaches. Proper noun dictionary of 5,000 words is separately designed apart from morphological dictionary of 30,000 words, the rules are not hard coded in the program and hence language independent and can be used for other languages by changing language specific aspects. The system has good recognition rate around 85%.

1.7.5 Implementation Named Entity Recognizer Using HMM

HMM is experimented. Hidden Markov Model (HMM) is a generative model. The HMM is denoted by **$\lambda = (A, B, \pi)$**. Where, **'A'** represents the transition probability. **'B'** represents emission probability and **'π'** represents the start probability. Each Hidden Markov Model is defined by states, state probabilities, transition probabilities, emission probabilities and initial probabilities. Words are observation symbols and POS tags are states in our HMMs. It is based on Markov Chain Property i.e. the probability of occurrence of the next state is dependent on previous state.

In order to define an HMM completely, the following three Elements have to be defined:

- Number of States (N) = Number of relevant Categories
- Number of Observation Symbols (M) = Number of Words of relevant categories in the language
- The initial state probability

  $\Pi_i = P\{q_1 = i\}$ …(1.1)

Where $1 \leq i \leq N$, q_1 is a category (state) starting a particular word group type.

- State transition probability

$a_{ij} = P\{q_{t+1} = j | q_t = i\}$ …(1.2)

Where $1 \leq i, j \leq N$ and qt denotes the category at time t and q_{t+1} denotes the category at time t+1.

- Observation or emission probability

  $b_j(k) = P\{o_t = v_k | q_t = j\}$ …(1.3)

Where $1 \leq j \leq N$, $1 \leq k \leq M$ and v_k denotes the k^th word and q_t denotes the current state. While building HMMs, a manually checked and certified tagged corpus is used for training. Tagged data of 2100 words is used for training. HMM is implemented using Java i.e. JDK 1.2 and neat beans ID Environment. Accuracy is around 93%.

1.7.6 Implementation of CRF Based Named Entity Recognizer

Conditional Random Fields (CRFs) are a probabilistic framework for labeling sequential data based on the conditional approach. The primary advantage of the CRF over the HMMs is the conditional nature, resulting in the relaxation of the independence assumption required by HMMs. The accuracy of classification is more in CRF approach due to addition of more features rather than joint probability alone as in HMM.

Conditional model is used to label an observation sequence $0 = (o_1, o_2, .o_T)$ by selecting the label sequence S= (s_1, s_2, s_T), that maximizes the conditional probability P*(S|O). Which is defined as below?*

$P\ (S/O)\ \alpha\ exp\ (\sum_{t=1}^{T} \sum_k \lambda_k \times f_k\ (s_{t-1}, s_t, o, t)$ … (1.5)

Where $f_k\ (s_{t-1}, s_t, o, t)$ is feature function, whose weight is learned via training. The value of feature function is either 0 or 1.

$$f_k(s_{t-1}, s_t, o, t) = \begin{cases} b(o,t) \text{ if } s_{t-1} = B\text{-}NEP \text{ and } s_t = I\text{-}NEP \\ 0 \text{ otherwise} \end{cases} \quad …(1.6)$$

$$b(o,t) = 1 \begin{cases} \text{if the observation sequence at position i = "NER"} \\ \text{0, otherwise.} \end{cases}$$

We have achieved higher accuracy in CRF approach as compared to the HMM model. The accuracy of classification is more in CRF approach due to addition of more features not like joint probability alone as in HMM.

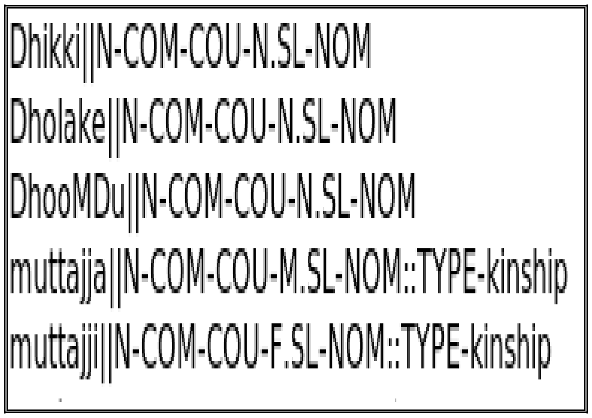
1.8 Main Contribution

Till now, no computational grammar exists for Kannada. In this workman exhaustive computational grammar for Kannada at word level is implemented by developing an automatic morphsyntactic annotator system. The most important aims of the contributed work are as follows:

i.) **Design and Development of Hierarchical POS tagset for Kannada,** which can be used for whole Dravidian group with modifications as required. The work started with the development of a Hierarchical Part of Speech (HPOS) tagset for Kannada as there was no standard hierarchical tagset available to perform the task of annotation. HPOS tagset captures detailed morpho-syntacic information of a word. A total of 170 atomic tag elements are used in developing the tag set. Kannada language is morphologically very rich in nature, the existing flat tag sets (IIITH, 2007) and AUKBC *tagset do not capture all the morph-syntactic features. So there is need for a tagset for Kannada which can capture the inherent variability and complex Kannada morphology.* HPOS provides a framework for a common Parts of Speech (POS) tagset for all Indian languages. Keeping this issue in mind, we have proposed a hierarchical tagset for Kannada, at the initial stage of our work. We have developed a set of annotating guidelines for Kannada and also for the whole of Dravidian group. There was no hierarchical tagset available when we started our work. Semantic information is also taken into consideration to some extent. The framework is designed by referring to Expert Advisory Group on Language Engineering Standards (EAGLES)as Guideline which is proposed by (Leech et al, 1996).**We have overcome many open issues that were not handled in IL-POST (Indian Language Parts of Speech Tagset) guideline proposed by Microsoft Research group (Baskaran et al., 2008),like handling of Clitic information, Transitive/Intransitive information on verbs which is useful information in subject object verb**

agreement, division of adjectives and adverb which is required in building chunking application, handling of conjunctive verbs and aspect auxiliary verbs which leads to formation of lakhs of verb forms.

ii.) **An electronic dictionary using a Hierarchical tag set is developed in the second stage of research.** Developing a dictionary using a hierarchical tagset is a tedious job since minute details of the word are to be marked for each word. Designing a dictionary using a hierarchical tag set is a*new attempt in the direction of developing dictionary and is first of its kind as compared to already existing flat tagset dictionaries*, which simply mention word category as n-noun, v-verb etc. Few e-dictionaries are available for Kannada like IIIT-Hyderabad's Kannada-Hindi dictionary, Prof. K.N Murthy's dictionary, online Kannada nighanTu are not directly usable for our work since treatment of words is different and there is lot of inconsistencies in these dictionaries and moreover words are tagged with a flat tagset. The existing situation made us develop a dictionary from scratch. Our dictionary stores Morph related information also which is useful for the Morphological generation process. The dictionary is developed using Pattern matching, filtering technique and frequency estimation techniques. The dictionary consists of 30,000 plus words. A Total of 126 rules are used for extraction of words from the Corpus, DoE, and CILL corpus (Department of Electronics, Central institute of Indian Languages) of size 3 million words is used to carry out the experiments. The words with irregular morphology are stored in the dictionary like locative nouns and a few inflected pronouns are stored as exceptions in the dictionary. We have used various combinations of pattern matching technique as mentioned in the previous section like stemming technique to get root words from corpus; our heuristic methods have produced a comprehensive dictionary of more than 30000 words. It is an ongoing work. Words are also added manually from different grammar books. Sample format of our dictionary is shown in figure 1.1.



Dhikki||N-COM-COU-N.SL-NOM
Dholake||N-COM-COU-N.SL-NOM
DhooMDu||N-COM-COU-N.SL-NOM
muttajja||N-COM-COU-M.SL-NOM::TYPE-kinship
muttajji||N-COM-COU-F.SL-NOM::TYPE-kinship

Figure 1.1. Format of Hierarchical Dictionary

Closed class words are function words and fixed in number. As of today 8898 closed class words are available in the dictionary, remaining 231621 words are open class words. We observe that existing dictionaries differ significantly in terms of the number and nature of grammatical categories and features they indicate. Thus we see that the available text corpora and electronic dictionaries are far from adequate and we need to develop a wide-coverage electronic dictionary giving detailed morph-syntactic information.

iii.) **A robust morphological analyzer/Generator (MAG) tool using finite state transducers (FST) technique is developed**, which uses the above developed resources like hierarchical tag set and dictionary. ***The development of the morphological analyzer and generator using hierarchical tagset is another new attempt in this direction. Aspects of Inflectional morphology, derivational morphology, external saMdhi are handled to the most extent.*** The performance accuracy of the system is more than 90 % for nouns and 100% for pronouns and around 81% for verbs.

iv.) **A rule based methodology to recognize Kannada named entities is developed.** Named Entities like name of a person, name of location, name of organization, number, measurement, and time are considered. Suffix, prefix list and proper noun dictionary of around 50, 00 words are developed in addition to 30,000 words morphological dictionary. Lack of capitalization features in Kannada make the NER task challenging. In Kannada proper nouns are indistinguishable from forms that act as common nouns and adjectives for example a city name like "Bangalore" is location name which is a noun as well as surname of person. This kind of ambiguity makes Kannada NER more challenging. The results of recognition are encouraging and the methodology has the precision and recall rate of around 86% and 87% respectively. The famous Kannada news paper PrajaavaaNi corpus is used to carry out experiments. *Rule based attempt for NER is the first attempt for Kannada Language.*

v.) **A statistical methodology using HMM for Kannada NER** Task is developed. Manually tagged data of 2100 words is used for training. The method gives accuracy of around 93%. HMM is implemented using Java, JDK 1.2 NET BEANSIDE, in Windows.

vi.) **Conditional Random Field (CRF) machine learning approach to recognize Kannada named entities is also attempted.** Conditional Random Fields (CRFs) are a probabilistic framework for labeling sequential data based on the conditional approach. The primary advantage of the CRF over the HMMs is the conditional nature, resulting in the relaxation of the independence assumption required by HMMs. The CRF approach uses more features than joint probability alone as in HMM.

1.9 Envisaged Applications

The following are the possible applications envisaged, depicting the usefulness of the Research work undertaken.

- The main function of this system is, to identify and enumerate all the construction types (within linguistic limits) of a particular language, down to all degrees of detail. This approach of construction labeling would be helpful in machine translations. The system is designed with the intention of being used for the MT system; however is not our focus now?
- The morphological analyzer/generator developed here can be used as a spell checker as well. Since morphological analyzer/generator give output morpheme by morpheme, showing the change of category, description of inflections in ordered manner depicting word formation process, hence the

morphological analyzer/generator module can also be used as a language learning tool.

- Another important issue in grammar engineering is the reusability of grammars. The more a grammar is committed to a certain processing model, the less are the chances that it can be adapted to other processing models or new application areas. This constitutes a serious bottleneck in the development of language technology products. Our morphology **module is language independent and can be adopted by any language just by replacing the rules of respective languages. There by reusing of system can be performed.**

- The lexical resources developed in this work like dictionary and NER provide means for developing tagged data. Tagged data is useful in carrying out statistical experiments.

- POS tagging acts as the base step in identifying chunks for IR information retrieval. POS tagging can be used as an intermediate step for higher NLP tasks such as Parsing, Semantic Analysis and Machine Translation.

1.10 Organization of the Thesis

The thesis is organized into 9 chapters. The subsequent chapters of the thesis are organized as follows. **Chapter 2** explains the literature survey. The literature survey is five fold since an attempt of developing five resources as components of annotator system is valued here. Issues regarding existing tagsets and their deficiencies are focused. In second stage, work on dictionaries is discussed. In the third stage POS taggers for Indian languages and also morphological analyzers is focused. Lastly survey on Named Entity Recognition is discussed. **Chapter 3** describes the work related to development of hierarchical POS tag set. Development of a full- fledged hierarchical tagset is a laborious task. Developing an exhaustive part of speech (POS) tagset covering linguistics features encoded in the words of the language is a major task. **Chapter 4** describes the needs for building an Electronic Dictionary using a hierarchical tagset, which was developed keeping morphological analyzer/generator tool as immediate application in the next stage of proposed work.

Chapter 5 describes about the methodology for developing Morphological Analyzer/Generator using finite state transducers. Morphological analyzer is developed covering inflectional and derivational features, external saMdhi and compounding. **Chapter 6** describes the development of Named Entity Recognizer, using the rule based approach. **Chapter 7**describes the Hidden Markov Model methodology for Kannada named entity recognition. HMM works on joint probability of observation sequence. **Chapter 8** describes CRF based machine learning approach for Kannada NER. Performance of CRF is good as compared to HMM. **Chapter 9** describes the conclusion of the work undertaken and the scope for future work, leading to possible improvements in developed methodologies.

APPENDIX A. Gives List of Hierarchical Tags. APPENDIX B. Gives Sample morph output. APPENDIX C. Sample Text Tagged with HPOS Tag set. APPENDIX D gives Sample Train Data for HMM. A detailed bibliography is provided at the end.

Chapter 2

Literature Survey

Chapter 1 focused on the problem definition and on a general perspective of the current scenario of research related to resources required for automatic morphosyntactic annotator system of Kannada. The AMAS architecture described here relies on resources like, tagset, and dictionary, Morphological Analyzer (MA) /Morphological Generator and Named Entity Recognizer (NER). None of these valuable resources existed for Kannada on which we could rely for proceeding our work. Hence development of five resources is focused here. Literature survey is therefore fivefold, i.e. the areas of survey work like tagset, dictionary, morphological analyzer/generator and NER are considered. The first section of the survey is regarding existing POS tagsets. The second section is regarding the survey of electronic dictionaries. The third section of the survey is regarding the morphological analyzer/generator and the fourth section is regarding named entity recognizer (NER). To know the state of art, literature survey is carried out. Following is the gist of the survey carried out.

2.1 Tagsets in Indian and Foreign Languages

Compilation of tagset is crucial for developing annotated corpora. Annotated corpora have drawn more attention of NLP researchers since the last decade. Lots of work has been carried out on the development of the tagset for English as compared to Indian languages. Following is the gist of works cited in the literature. The first attempt regarding the development of a hierarchical tag set was made by (Leech and Wilson, 1996). This system was generally referred to as Expert Advisory Group on Language Engineering Standards (EAGLES) standard. This system is adopted as the standard model by many researchers in developing a hierarchical tag set for their native Languages. EAGLES describe 11 major categories. The attributes are sorted out in a hierarchical structure in the form of intermediate tag sets. They proposed 114 tags for English and 274 for Italian.

(Hardier, 2004) has proposed a hierarchical tag set for Urdu. He has developed 280 tags for Urdu, in his research work viz. "Computational Analysis of Morphosyntactic categories in Urdu". The tagset discussed here was created in accordance with the EAGLES guidelines for morphosyntactic annotation of corpora. This design followed the description of Urdu grammar given by Schmidt in 1999 for the purpose of tagset design.

(Shereen Khoja et al., 2001) have proposed a tagset for morphosyntacic tagging of Arabic. They have devised 177 tags, 103 for nouns, 57 for verbs, 7 residuals and 1 for punctuation. This is considered an extended tagset which includes, voice modifiers, marking of transitive/intransitive features for verb and marking of derivation for nouns in the tag.

(Santorini, 1990) has proposed Part of Speech tagging guidelines for the Penn Treebank Project in English. The tagset is popular even today. It is flat tag set consist of 35 tags. (Garside, 1987) proposed a tagset for English. It is known as CLAWS (C5) tagset has 60 tags. Though these tagsets are developed for same the language they differ significantly from each other. The corpora tagged by these two tagset remain incompatible with each other due to lack of standardization.

Several POS tagsets have been designed for specific Indian languages by different research groups such as IIIT Hyderabad for all Indian languages, CALTS department of University of Hyderabad developed tagset for Telugu language and AU-KBC research centre for Tamil language.

(IIIT Hyderabad, 2007) has proposed tagset for all Indian languages. The tag set consists of 26 tags that capture only coarse level information. They are flat in nature, and do not include finer morph-syntactic features. Though it was intended as a common tag set for all Indian languages, the tags seem more suitable for Indo-Aryan languages like Hindi. This tag set is designed with an idea that lesser number of tags will result in efficient machine learning and reduce tagging errors in human annotation. Certain phenomenon has been handled in distinct ways, which can result in incompatibility across corpora. For example, in Hindi, postpositions and case-markers are written as separate words after modifying nouns. However, in Dravidian languages like Kannada case-markers are morphemes and are always suffixed to the nouns, whereas postpositions can be written separately or agglutinated with the preceding noun. The IIIT-H tagset tags the postpositions by assigning a separate tag as in Hindi like ko, ki etc, but when postpositions are suffixed to nouns as in the case of Dravidian languages, how it handles this situation is not clear. Consider the example below.

English: Ram killed a bird … (1)

Hindi : raam ne maaraa paMchi ko … (2)

Kannada: raama**nu** koMdanu pakshi**yannu** … (3)

For convenience, the word order is maintained as it is in the above examples. In case of English language, the position gives the roles played by Ram (subject) and bird (object). In case of Hindi, case markers *ne* and *ko* exist, but they do not inflect raam and raavana. In case of Kannada the subject raamanu is inflected for case prathama (nominative case ***nu*** as a suffix) and object pakshiyannu (bird) is inflected for accusative case (***annu as suffix***). In IIIT-H tagset ***ne, ko*** are postpositions separate tags are assigned, but IIIT-H tagset does not capture the case markers information when they are suffixed to nouns as shown for Kannada language in example 3 above.

(CALTS UoH, 2008) has proposed a tagset for Telugu. The tagset consists of 53 tags which are particularly designed for Telugu language. Though a finer distinction is made for each POS category, the order of distinction is not maintained. Gender for nouns is not taken into account. For example the structure of noun is ***noun+number (singular/plural) +case***. Person information, gender information for pronouns and gender for nouns is not considered. Separate tags are given for compound common nouns which is particularly useful for applications like machine translation.

(AU-KBC, 2001) has proposed a tagset for Tamil language. The tag set consists of 68 tags. It contains a large number of independent POS categories, however the tagset is flat in nature and it is designed exclusively for Tamil language only.

(SCIS, UoH) has proposed tagset for Telugu. The tagset is hierarchical in nature and used for chunking application using dependency parsing.

(Baskaran et. al, 2008) have proposed a common POS Tagset framework for Indian languages. It is generally referred to as IL-POST work has many issues left unsolved, like information on transitivity on verbs, clitics information, handling of aspect auxiliaries and modal auxiliaries etc. There is no subdivision in adjectives and adverbs; the subdivision of adjectives and adverbs is useful information in grouping chunks in chunking applications. Formation of conjunctive verbs is not handled. Contingent features of verbs not considered.

It is observed that existing POS tagsets for Indian languages offered by different research groups remain incompatible with each other in terms of morpho-syntactic categories and features and the tag definitions, levels of granularity are also different. The tagset design plays a vital role when data is tagged according to it and hence it affects the development of NLP tools within and across that language. These tag set captures word features only at a shallow level and tags can cover only shallow linguistic features ignoring linguistic richness of a language.

2.2 Survey of Electronic Dictionary

Dictionaries can be directly used by a variety of NLP applications such as spell checking, syntactic analysis, parsing, machine translation (MT) and morphological analysis etc. Wide coverage electronic dictionaries giving detailed morpho-syntactic information are essential and more valuable for developing POS taggers. Lots of work has been carried out on development of an electronic dictionary for English as compared to Indian languages. Following is the gist of works cited in the literature.

(Patrick Hanks, 2009) discuses the relationship between a lexical data base and monolingual dictionaries, the role of corpus evidence, need for vocabulary coverage. This article gives a survey of the main issues confronting the compilers of monolingual dictionaries in the age of the internet.

(Goldberg D. E, 1989) has proposed a genetic algorithm to fuse different linguistic hypotheses and also Machine Learning statistical methods to extract bilingual dictionaries. He used large amounts of parallel texts to build the dictionary.

These kinds of dictionaries are useful in Current corpus based machine Translation application.

(Tokunaga et al., 1995) have proposed automatic thesaurus construction based on grammatical relations among the words in the sentences. The proposed method constructs the thesaurus by using a hierarchical clustering algorithm. In the experiment four RBT (relation based thesaurus) of Japanese nouns were constructed from 26,023 verb-noun co-occurrences.

(Yamamoto, 1993) has proposed a method for generating a translation dictionary from Japanese /English parallel texts. In this method, English and Japanese compound noun phases are extracted from parallel texts and searched by matching their possible translations generated by the existing translation dictionary.

(Kumano A. and Hirakawa, 1994) have proposed a method of building Machine Translation dictionary from Parallel Texts Based on Linguistics and Statistical Information. This is useful in machine translation application.

(Kay M and Roscheisen M, 1993) have proposed a technique of Text-Translation Alignment using bilingual dictionary. They have presented an algorithm for aligning texts with their translations that is based only on internal evidence. The relaxation process rests on the notion of which word in one text corresponds to which word in the other text that is essentially based on the similarity of their distributions. It exploits a partial alignment of the word level to induce maximum likelihood alignment of the sentence level, which is in turn used, in the next iteration, to refine the word level estimate. The algorithm appears to converge to the correct sentence alignment in only a few iterations.

(Utsuro, Matsumoto and Nagao, 1994) has proposed a unified framework for bilingual text matching by combining existing hand-written bilingual dictionaries and statistical techniques. The process of bilingual text matching consists of two major steps, sentence alignment and structural matching of bilingual sentences.

Regarding Indian Scenario we have dictionary in Printed form like the Mysore University Kannada English dictionary. This dictionary is available in PDF form. It is not possible to process the required information by writing programs. It is meant for the end user to look up the meaning of an English word in Kannada and is not suitable for our work. For each English word, the corresponding explanation meaning or description is given in Kannada. As such, we do not always find equivalent Kannada words directly.

(IIITH -2007) Kannada-Hindi dictionary is another electronic dictionary available at IIIT –Hyderabad website. It is a bilingual dictionary which has both Kannada and Hindi words. IIIT-H Kannada-Hindi dictionary is in ISCII form. But it is advantageous as it is in electronic form which is a processable form and is useful to some extent in extracting head words by writing programs. But there is lot of inconsistencies. Many nouns are treated as avavyas which is not correct. Treatment of words differs from our principles. We cannot directly relay on this dictionary and moreover the dictionary is tagged with flat tagset.

(V. Krishna, p. c, 2009). This dictionary has 1 lakh words. It is in electronic form, developed using NuDi editor in Excel sheet file. Even though the dictionary has many words, they all belong to old Kannada literature, which have rare usage in modern Kannada literature. Moreover the words are tagged with a flat tagset. The tag does not capture detailed information. Any information useful for morphological process is not maintained in this dictionary.

(SCIS, UoH, p.c, 1997). The dictionary has around 10,000 entries. The number of POS categories used are 4. This dictionary has some inconsistencies say noun phrases are treated as nouns, this is not correct because a phrase consists of many words, for example the phrase “aMdaaju vaarshika paTTi” is marked as a noun. But the advantage is it has more words which are used in modern Kannada. The words however are tagged with a flat tag set.

We observe that a few online dictionaries like baraha nighaMTu, Kannnada-kasturi, and Prof Venkatasubbayya dictionaries are intended for getting corresponding English words for the given Kannada word. These kinds of dictionaries are not useful for our work. Ongoing projects related to corpus building and lexicography is not with standing, the importance and urgency of electronic lexicons in India cannot be overemphasized. Existing dictionaries need a lot of ironing, are not consistent enough, and do not match at top level category tagging. Words treated as nouns in one dictionary are treated as adjectives and adverbs in another dictionary. We cannot use these dictionaries for our work directly.

2.3 Survey of Morphological Analyzers

The mechanisms which derive new words, which analyze the existing words, are of the major concern to morphology. A list of basic units constitutes the lexicon of the language and specification of rules constitutes morphology of the language. The other part involving complex combinations of words into phrases and sentences is Syntax. Both morphology and syntax constitute the grammar of a language. In general, several approaches have been attempted for developing the morphological analyzer. Lots of work has been carried out on the development of morphological analyzers for Indian languages and also other foreign languages worldwide. Following is the gist of works cited in the literature.

(K. Koskenniemi, 1983) developed a two-level morphology approach, where he tested this formalism for Finnish language. This formalism introduces an additional surface level in order to deal with special morphological phenomena (example. vowel gradation) in an elegant and compact style. The model consists of two components, a linked lexicon component and a two level rules component. The two level rules component consists of morphological rules which describe the relations between the surface and the lexical forms.

(Kaplan and Kay,1994) showed how, manipulating regular languages and relations can provide a solid basis for computational phonology. Most of the accounts of such phenomena have used finite state technology. Finite state analysis of morphological and phonological phenomena for languages of complex morphology like Finnish and Turkish are discussed here. Later an extended attempt is presented regarding context-sensitive rewrite rules of the form $w \rightarrow W\lambda\rho$. Kaplan and Kay describe the morphological and phonological alternations in a natural language. The above rule states that the string w is replaced by the string W whenever it is preceded by the string λ and followed by the string ρ. Kaplan and Kay show that rewrite rules of this form define regular relations.

Furthermore, they show how these rules are compiled into finite state transducers. Thus, a single transducer representing a whole set of rules can be constructed from transducers which correspond to different phenomena by using the composition and union operators.

(Mohri, 1997) showed extended regular expression descriptions of languages and compilers of the expressions to finite state automata (FSAs) and transducers (FSTs). FS approaches for NLP have generally been very successful. It has long been claimed that the morphology of many languages lies within the expressiveness of a class of formal languages known as regular languages, and computational morphologists have made this claim.

(John Goldsmith, 2001) proposed a particularly interesting approach which can be seen as a special case of probabilistic idea. It is based on the Minimum Description Length (MDL) concept. The main intuition in this concept is that, if all the morphemes, which are the basic elements of all the words involved in an input, are assigned distinct numeric values in the smallest possible number of space, and then the input can be represented as a sequence of these numbers. The implementation of this approach is available as free downloadable software called Linguistica.

(Buckwalter, 2002, 2004a) proposed an Arabic Morphological Analyzer. The system consists of a lexicon and a Perl program implementing an original algorithm for recognizing inflected Arabic words. It is the most widely used tool of its kind. The coverage of the lexicon is excellent and the runtime performance of the program is very reasonable. Importantly enough, the first version of the Buckwalter analyzer was published as open source software. The analyzer consumes an ordinary Arabic text, resolves its contiguous orthographic strings, and produces morphological analyses characterizing each of them as a whole. The format of the output can be schematically depicted as follows: (composition of morphs).
[Lemma id] morph 1/tag 1 + morph 2/tag 2 + . . . + morph n/tag n.

(Hammarstrom et al., 2006) focused on unsupervised morphological analysis. Unsupervised approaches to morphological analysis (MA) are important for less studied (and corpus-poor) languages, where there is a small or no machine readable dictionary and tools. Ideally, an unsupervised morphological analyzer (UMA) would learn how to analyze a language just by looking at a large text in that language, without any additional resources, not even mentioning an expert or speaker of the language.

(Creutz, 2003) uses probabilistic distribution of morpheme length and frequency to rank induced morphemes. He outperforms Goldsmith for Finnish but gets worse results for English. Various approaches for unsupervised extraction of stems and suffixes have been reported for English.

(Dhanalakshmi et.al. 2009) Developed a finite state automata based morphological analyzer for Tamil language. Many attempts have been made in case of Bengali by and Marathi language morphology.

(T.N. Vikram and Shalini Urs, 2007) proposed a prototype for morphological analyzer for Kannada. This is just a prototype and moreover many features of morphology are not touched here. Derivational morphology is not handled. Auxiliaries are not handled, contingent features are not handled. Clitic morphology is also not handled.

(SCIS, UoH 1999) proposed a morphological component for Kannada as part of “A Network and process model tool for Kannada”, which handles only inflectional morphology. Size of dictionary used for morphological process is 10,000 words. Here derivational morphology is not handled.

(Umamaheshara Rao and Parameshwari, 2010) have done some work in respect of Kannada Morphology, but the approach is paradigm i.e. suffix list based. The tagset used by them is flat. Performance is limited by dictionary size. Clitic morphology information is not handled.

(ShaMbhavi, 2011) have attempted to build Kannada morphological analyzer and generator using trie and the approach is paradigm based. Derivational aspects, contingent features, causative voice modifiers, aspect auxiliaries are not discussed. Dictionary size is around just 3700 words only. Clitic morphology information is not handled. Formation of conjunctive verbs is not handled. The tagset used by them is flat, (Ramasamy and Soman K P, 2011) have developed a rule based morphological analyzer and generator, using finite state transducer. Handling of derivational morphology is not discussed. Here also many aspects of morphology are not reported. The tagset used by them is flat.

We observe that existing morphological analyzers for Kannada are not sufficient. Many of the features are not attempted like derivational morphology for noun, verb, adjective, compounding, handling of auxiliaries and both aspect and modal auxiliaries are also not reported in many systems, these limitations reduce the morphology based tagging performance. The tagset used by them is flat; the size of the dictionary they use for morphological analyzer/generator is not clearly mentioned. Many features like contingent, voice modifiers are not handled at all.

2.4 Survey of Named Entity Recognizers (NER)

Named Entity Recognition (NER) is a task of finding and classifying proper nouns like, person name, location name and organization name in the text. NER is an important task in many Natural Language Processing (NLP) applications like machine translation, question-answering systems, indexing for information or information retrieval, data classification and automatic summarization etc. NER has drawn more attention from NLP researchers since the last decade. Lots of work has been done on NER for English, employing machine learning techniques and also rule based approach. Modern systems most often use machine learning techniques because rule based approaches need more months of development by experienced linguists. However, handcrafted rule-based systems usually give good results. Following is the gist of works cited in the literature.

(Ralph Grishman et al., 1996) has developed a rule based NER system which uses some specialized name dictionaries including names of all countries, names of major cities, names of companies, common first names etc. It reports the recall, precision and f-measure as 86%, 90% and 88.19 % respectively. (Borthwick et al, 1998) have developed a Machine Learning based system. The system has used 8 dictionaries.

(Fleischman, 2001) has proposed a method for categorization of location names using Bayesian technique and the decision tree and the accuracy reported is 80%.

(Hsin Chen et.al, 2003) have proposed an algorithm for Named entity for Information retrieval. The different levels of text are employed, namely, character conditions, and statistical information. The recall rates and the precision rates for the extraction of person names, organization names, and location names are (87.33%, 82.33%), (76.67%, 79.33%) and (77.00%, 82.00%), respectively.

(Michael et al., 1999) has proposed ranking algorithms for Named Entity Extraction, using maximum entropy and reports precision, recall and F-measure as 84%, 86%, 85% respectively.

(Bick, 2004) has proposed rule based algorithm for Named Entity Recognition for Danish and reports F-measure as 92.7%.

(Ashara et al., 2003) has proposed a character based chunking algorithm for Japanese Named Entity Recognition and reported F-measure as 87.2%.

(Shilpi Srivastava et. al, 2011) have proposed a hybrid approach, a combination of rule based, and machine learning technique like condition random field (CRF) and maxEnt for named entity recognition are applied. The experiments are performed for Hindi Language and the hybrid approach reports precision of 96%, recall of 86.96% and f-measure of 91%. (Asif Exbal, 2008) has proposed a CRF based approach for Bengali and reports F-score of 89.3 %.

(Riaz, 2010) has proposed a rule based approach for Urdu, using small scale gazetteers and reports recall of 90%, F-measure of 93.14%, and precision of 96.4%.

(Amarappa and Sathyanarayana, 2012) came up with Named Entity Recognition and Classification for Kannada language, using SEMI-Automatic Statistical Machine Learning NLP models based on noun taggers using HMM. We observe that there is very little work on Kannada NER and no work is reported for Kannada NER using rule based approach. The English NER systems can not used directly for Kannada, since lack of capitalization features, lack of resources, and agglutinative property in Kannada pose challenges.

- **Summary**

From the literature survey it is clear that Kannada language is not sufficiently explored computationally as compared to English and other major languages of the world. Many Indian languages are lagging far behind in the field of NLP. There are hardly any substantial computational models for many of the Indian languages. It is clear that progress in Indian languages in terms of NLP technology has been very slow and Kannada is still lagging. Even today there is hardly any substantive computational part of speech (POS) annotator exists for many Indian languages.

Available dictionaries are not useful for the proposed tasks in hand. Existing dictionaries need lot of ironing and not consistent enough, and do not match at top level category tagging, words treated as nouns are treated as adjectives and adverbs in other dictionaries. These dictionaries are not directly usable. Morph related information is not captured in the already existing dictionaries.

Existing morphological analyzers/generators for Kannada are not sufficient. Many of the features are not attempted like derivational morphology for noun, verb, adjective, compounding, and handling of clitics, aspect auxiliaries and modal auxiliaries not handled. Un-handling of these features reduces the morphology based tagging performance. Inflectional morphology is also not handled to the extent required. Still exploration in this direction is required. *Use of hierarchical tag in building morphological system is not yet reported with respect to Indian languages.* Kannada needs to be explored in this direction.

It is also observed that there is very little work on Kannada NER and no work is reported for Kannada NER using rule based approach. NER aspects need to explore for Kannada language using machine learning techniques as well.

Chapter 3

Design and Development of Hierarchical Tagset for Kannada

From the literature survey it is observed that the design of a POS tagset for Kannada needs to be explored. The existing tag sets remain incompatible with each other in terms of morph-syntactic categories, tag definitions, levels of granularity etc. This chapter presents the first stage of research related to automatic morph-syntactic annotation for Kannada. Design of a hierarchical tagset based on syntax is proposed here. There is need of a good tagset for Kannada. The tagset plays a vital role in the development of NLP tools within and across languages. *The objective of a hierarchical part of speech tagset (HPOS) is, to capture the deep detailed information of word grammar which is necessary for many NLP applications. Another aim was to standardize the tagsets to achieve cross-linguistic compatibility, reusability and interchangeability.*

The tagset for Kannada has been designed considering the traditional grammar and lexical diversity. Some standard texts on Kannada Grammar were consulted and inputs from linguists were also sought. The tagged sentences were checked for suitability of parsing and information conservation. The tagset was enhanced based on the observations. In this way, iteratively a final tagset was designed. It is believed that the tag set is complete and sufficient for purpose proposed, but we are open to modifications and would like to receive constructive suggestions. Form of the word and function of the word are considered while designing the tagset. ***This is an exhaustive tagset for Kannada***. There was no workable POS (Part of Speech) tagset in existence for Kannada, when we started our work in 2007. POS tagger served as the fundamental building block for NLP research. EAGLES tag set is followed with modifications as required for our Kannada Language. The morphology of Kannada is complex, as compared to Turkish and Finnish. **The hierarchical tag set developed here can be adopted for the whole Dravidian language family.** ***The tagset is largely based on computational needs***. *We have compiled a tagset of 170 tags.* Compilation of tagset is basic and also an important task in all NLP applications.

Tagset designing becomes challenging for Languages like Kannada due to their agglutinative property. This chapter will look at the process of creating one of the necessary resources for the development of POS tagger for Kannada.

3.1 Why Categorization?

Categorization of words in a language is necessary since words have different grammatical roles in the sentence. Words in a language are classified based on the form and function roles and semantic and syntactic behavior. These word categories are called as syntactic or grammatical categories (Hardie, 2004). In a broad sense, words in natural language can be categorized as noun, verb, adjective, adverb etc. All the entities like people, animals and things are referred to as noun, actions involving motion or stative are referred to as verbs, entities acting as noun qualifiers are referred to as adjectives and entities acting as verb modifiers as adverbs. These categories are the syntactic category. Apart from these, another kind of categorization of words is made, known as functional words. Functional words are less semantic and few in number and are useful in binding the content words like noun, verb etc,. Functional words include preposition, conjunction, interjections. These are also known as 'structural words'.

3.1.1 What is a Tag?

Tag is a label representing grammatical information like verb, noun etc. Tagging or Annotation is the process of adding grammatical features like word category, case indicator, other morph features about the word in the text. The set of all these tags is called a tagset.

T**agset of higher granularity may actually gives better result**s. Annotated Kannada corpus is not available publically for use till today. **The first step for the annotation of corpora is the compilation of a tagset** that accurately describe and cover all the features of the language. So in the below section designing of hierarchical tagset for Kannada is explained.

3.2 Design and Development of Morphosyntactic Tagset for Kannada

Our Development of a tagset involves the following phases as shown in figure 3.1. Several design principles are adopted while designing the tagset. Section 3.2.1 explains the adopted principles. The Hierarchy of a proposed morphosyntacic tagset is shown in Figure 3.2. The methodology says that major word classes should be in the top level in the tree, followed by sub classifications and lastly morphological features. ***An issue of general concern is that in an effort to reduce the number of tags we should not miss out on the crucial information related to grammatical and other relevant linguistic knowledge which is encoded in a word.***

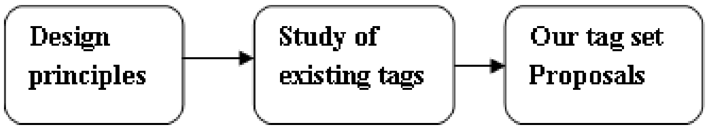


Figure 3.1. Design Phases in Development of Tags

3.2.1 Design Principles

The following design principles like granularity, form of a word and function of the word and semantic information that are considered while designing the tags are explained below.

i. **Granularity of tagset**

Granularity refers to capturing of minute details of the word. The tagset will be as fine grained as possible. The size of the tagset and its granularity are linked, the more fine grained analysis, the greater the number of tags must be. However (Veronis and Khouri 1995) points out, that the size of the tagset affects the performance of the taggers, particularly in probabilistic taggers, which require large training data. It is assumed that lower the granularity, the greater the accuracy. But it has been shown by researchers on the CRATER project that a tag set of higher granularity may actually give better results. Since tagset defined here is prior to any work on the actual tagger, it cannot be known what degree of fine grained tagset will prove optimal for the tagging process. Again it cannot be known in advance how distinctions will prove unfeasible. Therefore the first step must be to make a linguistically ideal tagset: The tagset which we would like to apply to our text in a real world, this ideal tagset, will be the largest within the parameters laid out by hierarchical design principles, on the basis that, it is always easier to remove distinctions than to add them. We propose the key design principle here of maximum granularity.

ii. Form versus Function

While designing the tagset, we were faced with the question, whether to tag the word based on its functioning or its form. Existing tagsets consider only word form. But we have tried to capture both form and functioning information while designing the tag. i.e. *if there is ambiguity in the word, say its lexical form is noun but functioning as a verb, then the word should be tagged with both tags.*

iii. Decomposability

A tag is considered to be decomposable if the string that it represents contains one or shorter strings or single characters that are meaningful out of the context of the original tag and may be found elsewhere in the tagset with the same meaning. For example, any noun is decomposable as “N-COM-COU-M.SL-NOM”, this tag is decomposable and is analyzed as N=noun, COM=common, COU=countable, M.SL=Masculine singular, NOM=nominative, the decomposable elements of the tag will indicate features in a hierarchy.

iv. Ease of modification

The hierarchical tag set is designed in such a way that if any new tag feature needs to be added in the future, it can be added easily without disturbing the whole tree structure.

v. Morphotactic rules

While designing the hierarchical tag set, the *order of suffix attachment to the word* is generally followed. As Kannada is a suffix based language, the list of suffixes is added in a particular order to the noun root and verb-root to form complex word forms. In case of nouns, first the number suffix (singular/plural) should be added followed by case suffix, later clitic suffix is added.

vi. Etymological Information

Etymology is the study of the history of words, their origins, and how their form and meaning have changed over time. No etymological information is considered in the tags.

vii. Syntactic Information

Whether to consider semantic information or syntactic information in tagset designing was a question. Existing tags have considered only syntactic information. We have tried to capture both pieces of information in the tagset design. For example syntactic information is marking of transitive, intransitive information for verbs. This information is useful to fulfill the complements demanded by verbs in parsing. Semantic information is also included to some extent like marking of countable and uncountable information and clitic information for nouns.

viii. No discourse information is included in the tagset.

The meaning associated with Discourse is judged at higher level in the sentence analysis during semantic analysis and is defined in terms of coherent sequences of sentences. Discourse aims at revealing socio-psychological characteristics of a person/persons rather than text structure.

ix. Tag for Multi token word

The multi token word like "Central institute of Indian Languages" will be tagged at individual token level. All linguistic features are encoded, hence there is no under specification or lack of features in the proposed tagset.

3.2.2 Study of Existing Indian Tag Sets

We observe that the existing tag sets are flat in nature and capture only the coarse level of information and are language specific. They need to be expanded when used for different applications like parsing or higher applications. The tag set proposed by (Baskaran et. al, 2008) is hierarchical in nature but has some deficiencies and have no sub classification of adverbs, adjectives and verbs. Clitics are missing.

The open issues of Baskaran's IL-POST tag are discussed in a further section. Issues regarding the tagset prepared by IIIT-Hyderabad (IITH, 2007) though it was intended as a common tagset for all Indian languages, the tagset seems more suitable for Indo-Aryan languages like Hindi. Certain phenomenon has been handled in distinct ways, which can result in incompatibility across corpora. For example, in Hindi, postpositions and case-markers are written as separate words after modified nouns. However, *in a Dravidian language like Kannada, case-markers are morphemes and are always suffixed to the nouns how IIIT-H tagset handles this situation is not clear.* The CALTS department tag set for Telugu and other flat tag sets are already discussed in chapter 2.

3.2.2.1 Comparison of Flat Tags vs. Hierarchical Tags

Flat tagsets are those which just list down mutually exclusive categories without any provision for extensibility, modularity and decomposability. This disparity hinders interoperability and reusability of annotated corpora. This affects NLP research in resource poor Indian languages where non availability of data, especially tagged data, remains a critical issue for researchers. Moreover these tagsets are flat in structure, capturing morphosyntactic features only at a shallow level. *Flat tagsets cannot capture finer language-specific properties because of their rigid nature*. So they cannot be used for annotating across a group of languages. But hierarchical tagsets are structured relative to one another. A Hierarchical tagset has a tree like extensible structure. The EAGLES guidelines were based on this concept of hierarchical tagsets. *The morphosyntactic details are encoded in separate layers of hierarchy. Major categories are at the top level and as we progress down the tree morphosyntactic features are covered in detail. The structure of our tagset is as follows: main category, subcategory, and sub-sub-category type and features.* Flat tagsets do not have *modularity and extensible capability.* But the hierarchical arrangement in *hierarchical tag set allows for selective inclusion and removal of features*. This is another advantage. Decomposability is becoming a desirable feature in the design of tagsets and it allows different features to be encoded in a tag by separate sub-strings. Comparison of flat versus Hierarchical tag set is listed in table 3.1.

Table 3.1. Comparison of Flat Tag versus Hierarchical Tag Set

Features	**Flat Tag Set**	**Hierarchical Tag Set**
Structure	Flat	Tree like
Extensible	No	Yes
Subcategory	No	Yes
Modularity	No	Yes
Decomposability	No	Yes
Size of tag	Small	Large
No. of Tags	Fewer	More
Information Capture	Coarse	Fine grained

3.3 Main Categories in Hierarchical POS Tagset (HPOS)

There are mainly 10 categories in the top level. The hierarchical POS tag set is designed by studying various grammar books like “A Kanarese Grammar” by Harold Spencer in 1950, “The grammar for spoken Kannada” by Harold Schiffman 1979. Another book “A grammar of the Kannada language in English” by Kittel in 1903. “A Descriptive Grammar of Kannada language” by S.N. Sridhar. The categories in hierarchical tag set are shown in figure 3.2.

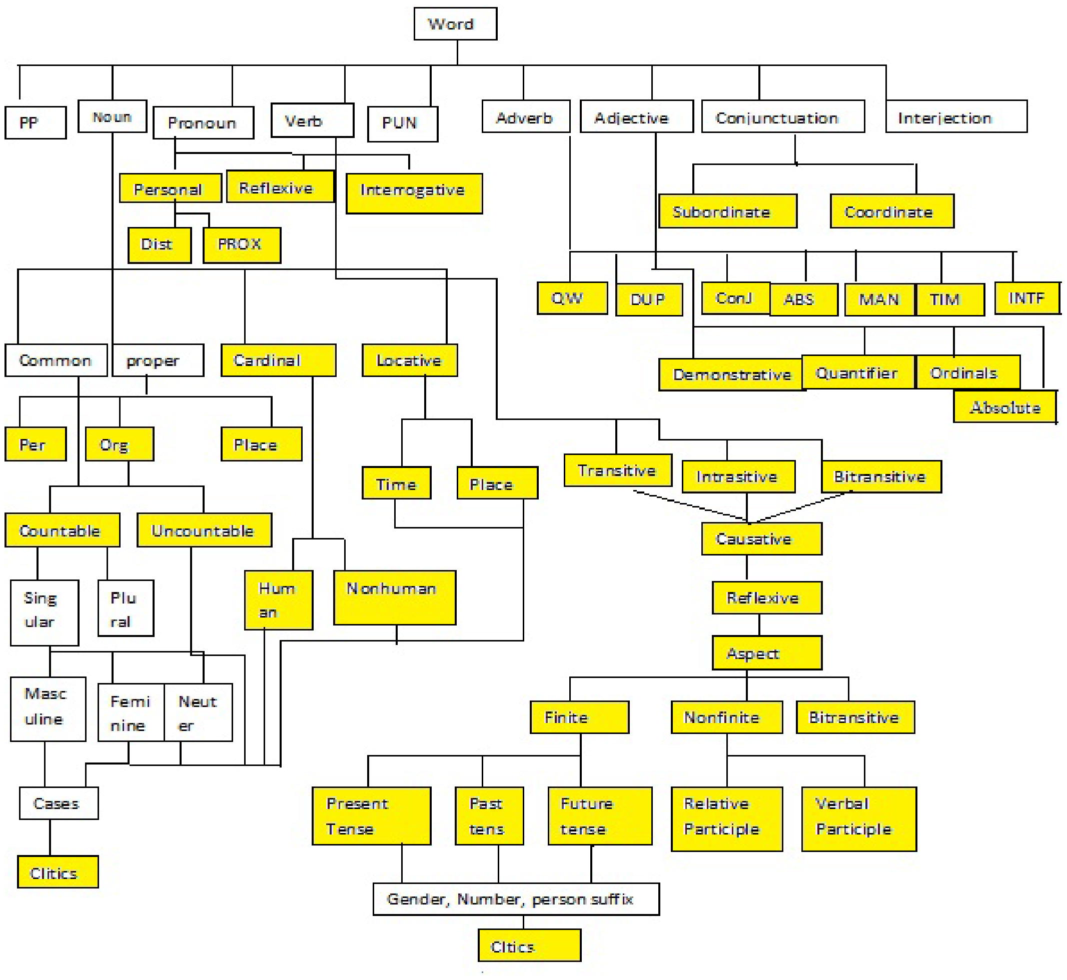


Figure 3.2. Hierarchy of POS Categaries

Each category is represented as a combination of uppercase letter; maximum length of tag is 4 letters. Word categories are systematically related by morphological process, like formation of plural, common inflections etc. The different categories their general structure and the sub categorization are shown in table 3.2 below.

Table 3.2. List of HPOS Basic Morphosyntactic Categories with Tag

Category	Tag	Sub Type	Sub-Sub-Type	General structure
Noun ನಾಮಪದ Naamapada	N	Common	Countable	N-TYPE-SUBTYPE-CASE-CLITIC
			Uncountable	
		Proper	Person	
			Location	
			Organization	
		Locative	Time	
			Place	
		Cardinals	Human	
			Non Human	
Pronoun ಸರ್ವನಾಮ Sarvanaama	PRO	Personal	Distance	PRO-TYPE GNP-DIST/PROX-CASE-CLITIC
			Proximate	
		Interrogative		
		Reflexive		
Adjective ವಿಶೇಷಣ visheshaNa	ADJ	Demonstrative		ADJ-DEM
		Quantifiers		ADJ-QNTF
		Ordinals		ADJ-ORD
		Absolute		ADJ-ABS
Adverb ಕ್ರಿಯಾವಿಶೇಷಣ kriyavisheshaNa	ADV	Time		ADV-TIM
		Place		ADV-PLA
		Manner		ADV-MAN
		Reduplication		ADV-DUP
		Question		ADV-QW
		Intensifier		ADV-INTF
		Absolute		ADV-ABS
		Conjunction		ADV-CONJ
Conjunction	CONJ	Subordinating		CONJ-SUB

ಸಂಯೋಗ saMyoga		Coordinating		CONJ-CORD
Interjection ಉದ್ಗಾರವಾಚಕ udgaravaacakagaLu	INTJ			
Postposition ಪದಪುಂಜ padapuMja	PP	Genitive Accusative Comparative Purposive Similarities Associative Others		PP-TYPE
Punctuation Mark ವಿರಾಮಚಿಹ್ನೆಗಳು viraamcinhegaLu	PUN	Separating Mark Quotation Mark Bracket	;,.”’[]{}…()	PUN-TYPE
Verbs ಕ್ರಿಯಾಪದ kriyapadagaLu	V	Transitive Intransitive Bitransitive	Finite Nonfinite Infinitive	V-TR/IN/BI-CAU-REF-MA-RP-INF V-FI-CAU-REF-MA-Tense-person suffix
Unknown	UNK	-	-	-

3.3.1 Nouns

Nouns refer to entities in like people, animals, and things. The general structure for noun considered is ***root+ gender (M/F/N) +number (SL/PL) +case + clitics***. Characteristics of noun are considered here are as follows.

i. A noun can modify another noun.
ii. A noun is the head of noun phrase.
iii. A noun can be inflected by case markers, clitics.
iv. Nouns in genitive case are treated as adjectives.
v. A noun can replace a pronoun.

vi. A noun is be modified by adjective. Adjective can come before a noun as noun modifier.

vii. Nouns and Pronouns share common inflections like case, clitics. The general inflections of nouns are listed in table 3.3.

Table 3.3. Common Inflections for Nouns

Type of inflection	Description
Number	Singular[SL], Plural[PL], Honorific[HON]
Gender	Feminine, Masculine, Neuter
Person	1 ST PERSON[P1] , 2ND Person[P2], 3rd Person[P3
Case	Nominative[NOM], Accusative[ACC], Genitive[GEN], Dative[DAT], Ablative[ABL], Locative Dative2[LOCD2] Comparative[COMP],Locative-Dative[LOCD1], Locative[LOC], Locative2[LOC2],Purposive[PUR1], Purposive2[PUR2], Sociative [SOC], Sociative2[SOC2]
Clitics	Emphatic[EMP],Interrogative[INTR],Inclusive[INCL], Indefinite[INDF]

All the linguistic aspects are considered while designing a hierarchical tagset for noun. Nouns are further classified as

i. Common noun-COM.

ii. Propernoun-PRP.

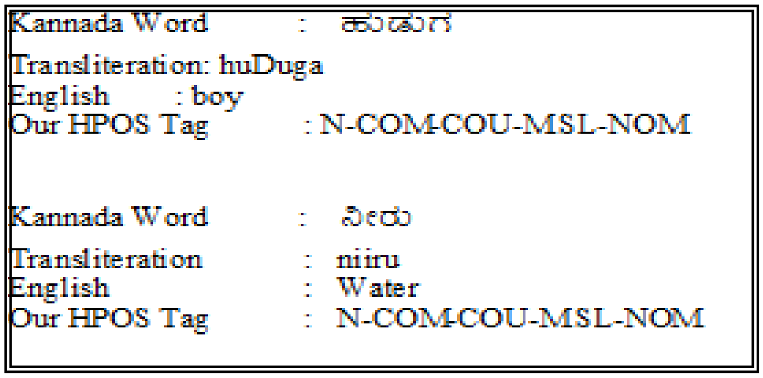
iii. Cardinal noun -CARD.

iv. Locative noun –LOC.

i. Common Noun - COM

The structure of common countable noun is ***root+gender+number (singular/plural) + case + clitics.*** There is no concept of person in common countable noun. Gender can be masculine, feminine or neutral. Since there are 3 genders and 2 numbers the total tag categories possible in the second level are 6. They are M.SL, F.SL, N.SL, M.PL, F.PL, and N.PL. In the third level case gets added to each of the categories present in the second level. In the fifth level 4 clitics are added.

Interrogative (aa), Inclusive (uu), Indefinite (oo), Emphatic (ee). In the sixth level interrogative clitic “a” is added to all clitics in the previous level. The common nouns are still decomposed into countable and uncountable, this distinction is necessary since uncountable nouns are not inflected for plural for example the word ***niiru*** “water” has no plural form **niirugaLu (waters)**. All such common nounsproperties are to be distinguished as uncountable. This type of finer distinction is useful for further high level application like parsing in subject verb agreement. Hierarchical tags for countable noun boy and uncountable noun water are shown below in the example in box 3.1 for these words.



Kannada Word	: ಹುಡುಗ
Transliteration	: huDuga
English	: boy
Our HPOS Tag	: N-COM-COU-MSL-NOM
Kannada Word	: ನೀರು
Transliteration	: niiru
English	: Water
Our HPOS Tag	: N-COM-COU-MSL-NOM

Box 3.1. Countable and Uncountable Word Tags

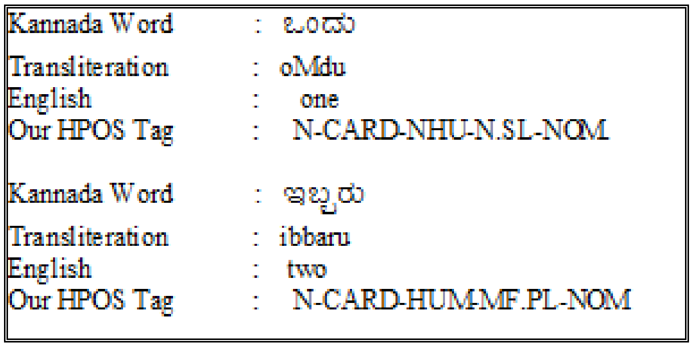
The tag is described as noun, common, uncountable, neuter singular. N.SL is a tag element with two atoms indicating N for neuter SL for singular.’-’ marks the hierarchy or subcategories.

ii. Proper Noun – PRP

The structure of a proper noun is proper ***noun +subtype e+ gender+ case+ clitics.***There is no concept of person in proper nouns. Proper noun is decomposed into proper noun indicating location-LOC, organization-ORG, and person-PER. Proper nouns are not modified by adjectives.

iii. Cardinal – CARD

The structure of noun cardinal is ***root+subtype+gender+number +case+clitics.*** Cardinals are special type of nouns in Kannada. Any word denoting number is tagged as cardinal. Cardinals are not modified by adjectives. Cardinals are decomposed into nonhuman-(NHU) and human-(HUM). Noun cardinals like eraDu (two), muuru (three) give plural sense but tagged as neuter singular. Consider the cardinal examples as shown in box 3.2.



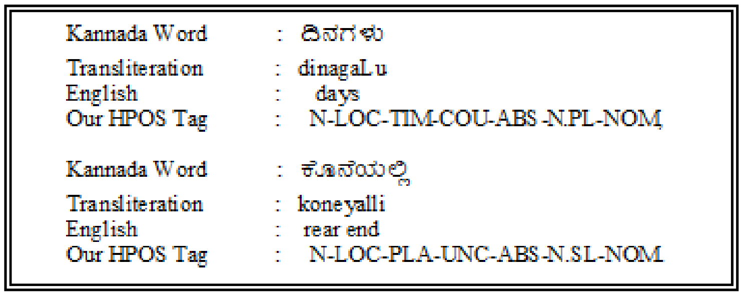
Kannada Word	:	ಒಂದು
Transliteration	:	oMdu
English	:	one
Our HPOS Tag	:	N-CARD-NHU-N.SL-NOM
Kannada Word	:	ಇಬ್ಬರು
Transliteration	:	ibbaru
English	:	two
Our HPOS Tag	:	N-CARD-HUM-MF.PL-NOM

Box 3.2. Tag for Human and Non Human Cardinals

The tag elementMF.PL indicates masculine or feminine in plural sense.

iv. Locative Nouns - LOC

Locative nouns are special type of nouns. They cannot be the head of a noun phrase. Locative nouns are again classified with respect to time and place. These words always indicate the location or time with respect to an entity, person and action. They take different lexical categories depending upon the context. Consider the examples shown in box 3.3.



Kannada Word	:	ದಿನಗಳು
Transliteration	:	dinagaLu
English	:	days
Our HPOS Tag	:	N-LOC-TIM-COU-ABS-N.PL-NOM
Kannada Word	:	ಕೊನೆಯಲ್ಲಿ
Transliteration	:	koneyalli
English	:	rear end
Our HPOS Tag	:	N-LOC-PLA-UNC-ABS-N.SL-NOM

Box 3.3. Tag for Time and Place Locative Nouns

The structure of locative noun is as follows. ***root+ subtype +subsubtype +feature + numbe r+ case+ clitics.*** This classification is required at full parsing level for indication of time and place modifiers. Some locatives act as adverbs and postposition also. Our assumption is that if case gets added to the noun then include it in locative nouns otherwise treat them as either adverbs or postposition. Derivation of nouns from verbs and adjectives are handled in derivation morphology process.

3.3.2 Pronouns -PRO

The structure of a pronoun is as follows. ***Root+gender (M/F/N/MF) +number (SL/PL) +person(P1/P2/P3) +case+clitics.*** Basic rules for pronouns considered here are as below.

i. A pronoun can be replaced by a noun or pronoun acts in place of a noun.
ii. A pronoun can be head of a noun phrase.
iii. A pronoun can be inflected by case markers and clitics.
iv. A pronoun cannot be modified by an adjective. An adjective is a noun modifier but not a pronoun modifier. For example consider the sequence of words. The adjective 'beautiful' (suMdara) follows pronouns avaLu/avanu (he/she) but does not precede the pronoun. (suMdara avanu "beautiful he" is not grammatically admissible).
v. All pronouns in genitive case are adjectives.
vi. Pronoun cannot modify a noun or another pronoun.

Pronouns are mainly classified into 3 categories.

i. Interrogative pronoun
ii. Personal pronoun
iii. Reflexive pronoun.

All the pronouns share features like case inflections, gender and number.

i. Personal Pronoun -PER-PRO

The structure of a personal pronoun is as follows. ***root+person (P1/P2/P3) +number (SL/PL)+distant/proximate (DIST/PROX)+case+clinics.*** In personal pronouns gender is not applicable in case of 1st person and 2nd person whereas in 3rd person it is considered. The total categories possible in the next level are P1.SL, P1.PL, P2.SL, P2.PL, P3.M.SL, P3.F.SL, P3.N.SL, P3.MF.PL, and P3.N.PL.For third person categories a further distinction is made to denote distinct and proximity information. The Hierarchical tag for personal pronoun avanu (*he*) is PRO-PER-DIST-M.SL-NOM.

ii. iInterrogative Pronouns -PRO-INTG

Interrogative pronouns exhibit another kind of wh-words, like who, which etc. All interrogative pronouns are of third person. Gender information is also considered. The structure of the interrogative pronoun is.***root+type+person (P3) +gender (M/F/N/MF)+number(SL/PL) +case+clitics.*** Hierarchical tag for interrogative pronoun ***yaaru*** *(who) is PRO-INTG-P3.MF.PL-NOM.*

iii. Reflexive Pronouns -PRO-REFL

The structure of the reflexive pronoun is ***root+type+person (P2, P3) +gender (MFN) +number (SL/PL) +cases+clitics***. The Hierarchical tag for reflexive pronoun *taanu (self) is PRO-REFL-P23.MFN.SL-NOM.*

3.3.3 Postpositions– PP

The postpositions in Kannada are similar to prepositions in English; these are small words that prototypically express spatial relations. Dravidian languages are morphologically rich in nature and postpositions occur as bound suffixes and some postpositions occur as free suffixes also. But in English and Hindi they occur as free morphemes. The suffixes like oDane, oTTige, oMdige (all give the English meaning *with*) occur as free suffixes apart from occurring as bound suffix also. Postpositions include case marker suffixes like nominative, genitive, dative, accusative, ablative, comparative, purposive, locative, similaritive, and locative-dative. Case marker suffixes usually occur as bound suffixes with nouns, pronouns etc. The postpositions which occur as free morphemes are tagged as PP-OTH; this tag stands for postpositions of other categories.

3.3.4 Adjectives -ADJ

Adjectives belong to the category of words which modify nouns. Adjectives cannot be head of the noun phrase and adjectives are not inflected for cases clitics. In Kannada clitics are added to all categories except adjectives. One adjective type can be modified by another adjective type. Say basic/absolute adjectives are modified by quantifiers. Adjectives are mainly classified into 4 categories.

i. Demonstrative adjectives .
ii. Quantifiers.
iii. Ordinals.
iv. Absolute adjectives.

In Kannada we can derive new adjectives from nouns and verbs also. There are productive rules in Kannada to derive adjectives. This part of derivation is handled in our morphological module in chapter 5. Bounded adjectives consist of all words belonging to color, taste and density. Adjectival quantifiers consist of words referring to quantity or size. Ordinals are adjectives derived from noun numerals (cardinals). The total number of tags possible under adjectives is 5.

i. Demonstratives -DEM

Demonstrative adjectives are those words that can be used only as adjectives and not as anything else, and they always appear in the adjectival position, that is, immediately before the nouns which they qualify. Demonstratives are few in number. Words like **ಆ** (that),**ಈ** (this) are used as demonstrative adjectives. These are used in the sense of "that boy", but not in the sense of personal pronouns. Consider an example for demonstrative pronoun.

Kannada	:	**ಆ ಮನೆ ಸುಂದರವಾಗಿದೆ.**
Transliteration	:	aa mane suMdaravaagide.
English	:	That house is beautiful

Here **ಆ,** "aa", (that) is tagged as ADJ-DEM.

ii. Quantifier –QNTF

Adjectival quantifiers consist of words referring to quantity or size. Words like bahaLa (much), svalpa (little) act as noun modifiers. We have kept it under subtype in adjective. Consider an example of a quantifier: *bahaLa "much" is tagged as ADJ-QNTF*. Some quantifiers like ashTu (that much), ishTu (this much) are quantifiers and are marked for inflection, which violate the theory that, adjectives are not marked for inflections. How to handle such words was a question: later we decided to store such forms under noun cardinal non human category.

iii. Ordinals –ORD

Ordinals denote quantifiers like first, second, third, in Kannada these words are derived by adding 'eeya' to cardinals oMdu (one), eraDu (two)etc. These behave as adjectives modifying nouns hence placed in adjective. Consider the example. **ಒಂದನೇಯ** *oMdaneeya (first) is tagged as ADJ-ORD.* Ordinals can be conjugated with third person personal pronoun suffix like avanu, avaLu, adu, avaru, avu to form nouns like **oMdaneeyavanu** etc. this is a productive rule to derive a noun from an adjective.

iv. Absolute –ABS

Absolute adjectives are used to describe the properties of nouns. These include basic adjectives like color adjectives and true adjectives like haLeya (old), hosa (new) etc. **ಹಳೆಯ**–(haLeya), "old" is tagged as ADJ-ABS.

3.3.5 Adverbs -ADV

Adverbs act as modifiers for verbs and they can be moved anywhere in the sentence. We have decomposed adverbs into 6 subtype categories.

i. Adverb (Absolute [ABS]) Example: bahus'aH (mostly) is tagged as ADV-ABS
ii. Adverb of Place [PLA]. Example: atta (that side) is tagged as ADV-PLA
iii. Adverb of Time [TIM]. Example: begane (early) is tagged as ADV-TIM
iv. Adverb of Manner [MAN]. Example: jooraagi (fast) is tagged as ADV-MAN
v. Adverb of Duplicate [DUP].Example: Sarasarane is tagged as (ADV-DUP)
vi. Adverb of Question [QW].Example: eenu (what) is tagged as (ADV- QW)
vii. Adverb (Intensifier [INTF]). Example: ati (too much) is tagged as (ADV-INTF)
viii. Adverb of Conjunction. Example: eMbudaagi (ADV-CONJ)

In Kannada we don't have a distinction as comparative and superlative adverbs, unlike in English say taller/tallest, smaller/smallest etc. Certain adverbs like ಅತಿ "ati" (too much) are specialized to the role of modifying adjectives and adverbs also. Adverbs of manner indicate the manner in which something is done. Derived adverbs are those which are derived from nouns and adjectives. The derivation morphology is performed by the addition of suffixes like aagi, aane, ge to nouns. Suffix "**aagi"** is used derive to adverbs from adjectives. Adverbs of time, place, and manner can be reduplicated to give specialized meanings such as repetition of event, emphasis to form adverbs of reduplication. Adverbs of manner (derived from nouns and adjectives) form an open class words. Adverbs of time, place, reduplication, negative, question and intensifiers and other manner-adverbs form a closed class of words. Total number of tags possible under adverbs is 11.

- **Adverb of Conjunction ADV-CONJ**

Two main clauses may be combined into one sentence using a coordinating conjunction. Alternatively, there may be two different sentences with the second sentence starting with a conjunction. At the level of syntax, since only one sentence is processed at a time, these conjunctions must be treated as adverbs. Consider different example sentences:

a) raamanu cennaagi haaDidanu ***aadare*** oLLeya phalitaaMs'a baralilla. Here ***'aadare'*** is taken as adverb of conjunction and tagged as ADV-CONJ.

- **Adverb of Duplication - ADV-DUP**

Duplications in adverb are referred to as onomatopoeic it is a type of manner adverb which gives more emphasis. Say in sarasarane (quickly - quickly) the first word is duplicated. Many of the tags proposed above have no detailed distinctions for adverb. Handling of onomatopoeic forms like sarasarane, barabarane which indicate emphasis are not indicated. We have included the finer distinctions. It is necessary to know occurrence of adjectives and adverb in the sentence. Which adverb follows which adverb, is a useful information chunking.

3.3.6 Interjection - INTJ

All categories except interjection have subcategories. Interjection refers to that part of speech which expresses emotion or a reaction, usually exclamatory emotions.

Interjections have no other grammatical relationship to the rest of the sentence. Interjections form a class of closed class words in Kannada. There is no further classification for interjection. Consider an example; say **ವಾರ್ರೆವಾ–** (vaarevaah) the tag is INTJ.

3.3.7 Punctuation marks -PUN

The punctuation marks are used to clarify meaning by indicating aggregation of words into sentences and clauses and phrases. Punctuation marks can be classified into general separating mark, quotation mark, brackets (left and right) in the first level. Left brackets are again divided into angular bracket, flower bracket, square bracket and parenthesis. Similarly the right brackets. Quotation marks are again divided into single quotes and double quotes. As there are 7 separating marks a separate tag is given for each of them. Punctuation marks form a closed class words in Kannada. The total number of tags possible under punctuation marks is 17.

3.3.8 Conjunctions -CONJ

Conjunction is a different type of word category which conjoins phrases or sentences. Coordinating conjunction conjoins or coordinates two words or phrases of the same category. Conjunctions are classified into coordinating and subordinating.

i. Coordinating Conjunctions –CORD Coordinating conjunctions refer to "and/or/but/therefore/so of English.
ii. Subordinating Conjunctions -SUB

Refer to "that" in English. The difference between coordinating and subordinating conjunction is, coordinating joins two sentences as equals whereas subordinating attaches a secondary sentence to a primary sentence. The secondary sentences often express a proposition, a reason, a condition, a concession, or a temporal event. The total number of tags possible under conjunctions is 2. Consider the behavior of conjunction examples in box 3.4.

Kannada : ರಾಮ ಮತ್ತು ಸೀತೆ_(1) Conjunction joining two Nouns
Transliteration : raama mattu siite
English : raama and siitaa

Kannada :ಅವಳು ಬಿದ್ದಳು ಅಥವಾ ಎದ್ದಳು... (2) Conjunction joining 2 verbs
Transliteration : avaLu biddaLuathava eddaLu
English : She fell down or got up

Kannada : ಹಸಿರು ಬಟ್ಟೆ ಮತ್ತು ನೀಲಿ ಬಟ್ಟೆ… (3)Conjunction joining two noun Phrases.
Transliteration : hasiru baTTe mattu niili baTTe
English : Green Cloth and Blue Cloth

Kannada : ಅವಳು ಓದಿ ಉತ್ತಿರ್ಣಳಾದಳು ಆದಕ್ಕೆ ಅವರು ಸಂತೋಷಪಟ್ಟರು...(4)
Transliteration: avaLu oodi uttiraNaLadaLu adakke avaru saMtoshapaTTaru
English : She read passed hence they felt happy

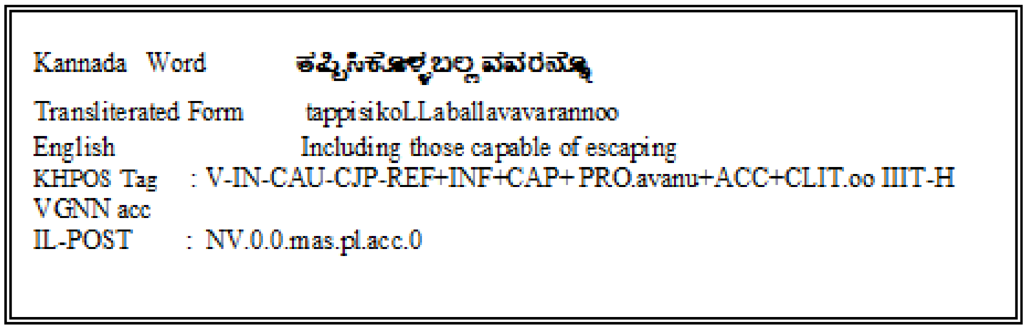
Conjunction joining two Sentences.

Box 3.4. Examples on Conjunction Behavior

3.3.9 Verbs

Verbs in Kannada have a complex morphological structure. We have tried to capture all the verb features in our verb tag set. The tag set is hierarchical in nature and captures detailed information in decomposable elements which constitute as sub strings of the verb tag. Verbs are the most important component of any sentence. These words talk about the action or the state of any noun or subject in the sentence. This means that verbs show what the subject is doing or what is the state or situation of the subject. A regular English verb has only one principal part, from which all the forms of the verb can be derived. This is the base form that is listed in dictionary. For example the verb *exists in English has,* the inflected forms like (*exists*, *existed*, *existing*).

Actions involving motion or stative are referred as verbs. Kannada verbs are very complex. Verb formation in Kannada is complicated. For the above verb exists (**ಇರು,** "be") we have more than 3000 different verb forms in Kannada one can observe the complexity involved in Kannada language. We can add any number of affixes to form complex verb root. A single root word in Kannada can give rise to a very large number of its surface forms. The richness of Kannada lies in the fact that significant part of grammar is handled by word morphology. Consider the formation of complex verb in Kannada.



Kannada Word ತಪ್ಪಿಸಿಕೊಳ್ಳಬಲ್ಲವವರನ್ನೊ
Transliterated Form tappisikoLLaballavavarannoo
English Including those capable of escaping
KHPOS Tag : V-IN-CAU-CJP-REF+INF+CAP+ PRO.avanu+ACC+CLIT.oo IIIT-H VGNN acc
IL-POST : NV.0.0.mas.pl.acc.0

Box 3.5. Tag for Complex Kannada Verb

The above Kannada word form is derived by adding 9 suffixes to the root verb ತಪ್ಪು–"tappu" (wrong). The formation of this verb form is Verb-intransitive+causative+verbal participle+reflexive + infinitive +modal (CAP) +PRO.avaru+accusative+clitic indefinite. The single word is equivalent to sentence in English as shown given box. IIIT-H Tags for above word is: VGNN acc (gerund accusative case) detail information is not captured here, the tag VGNN stands for gerunds. IIIT-Hyderabad tag set though it was intended as a common tagset for all Indian languages, the tags seem more suitable for Indo-Aryan languages like Hindi. In IL-POST tag for above derived word is verbal noun masculine and accusative case. In applications like Machine Translation tags capturing more detailed information are useful. So designing the verb tagset is a challenging task. We have compiled 42 tags for Kannada verbs. This is an exhaustive tag set existed so far as far as Kannada is considered. The information regarding aspect auxiliaries, modal auxiliaries and a special contingent feature of Kannada and a set of defective verbs are handled in this work.

Kannada is a verb final language. Verbs occupy first level in the tree. Verbs are used to describe actions, activities and state. The verb tags are proposed based on tree hierarchy is shown in figure 3.3. The main aim here is to develop tags for all possible inflections that occur for the verb.

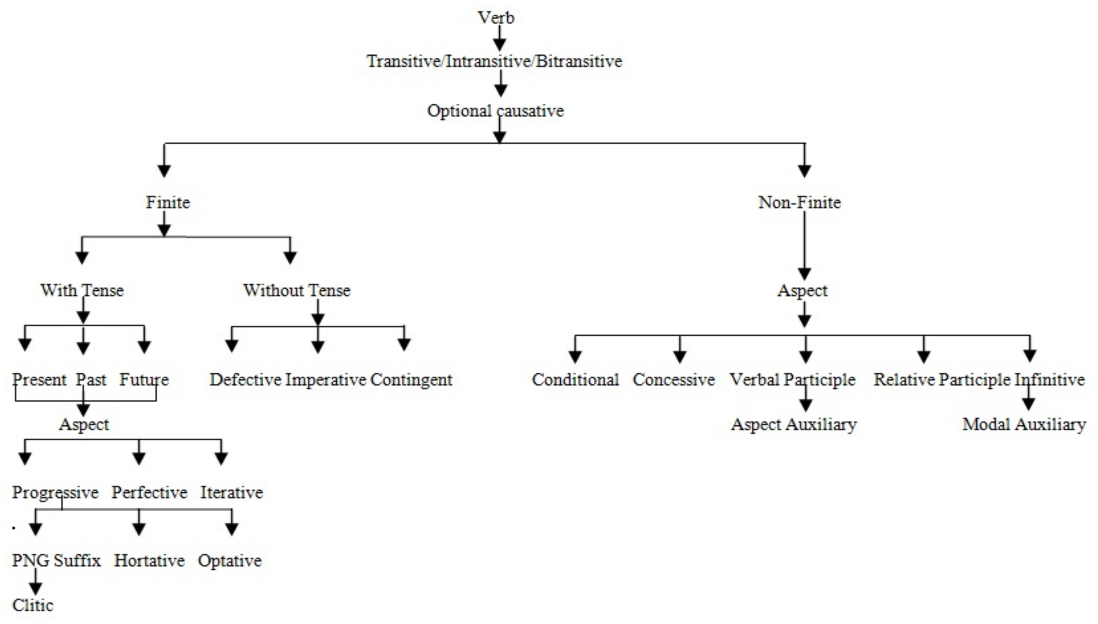

Verb
Transitive/Intransitive/Bitransitive
Optional causative
Finite
Non-Finite
With Tense
Without Tense
Present Past Future
Defective Imperative Contingent
Aspect
Progressive Perfective Iterative
PNG Suffix Hortative Optative
Clitic
Aspect
Conditional Concessive Verbal Participle Relative Participle Infinitive
Aspect Auxiliary
Modal Auxiliary

Figure 3.3. Verb Hierarchical Tree

The general tag structure for Verb – (V) is as follows

Verb Types	Voice Modifier	Finite	Non Finite

Box 3.6. General Structure of Verb Tag

3.3.9.1 Transitive/Intransitive/Bi-transitive Verbs

Verbs are classified as transitive (TR), intransitive (IN) and bi-transitive (BI) based on type of argument, that the verb takes.

i. Transitive -TR: Transitive verb is one which requires an object. Transitivity is the number of objects a verb requires or takes in a given instance.
ii. Intransitive -IN: Intransitive verb is one which does not require an object
iii. Bitransitive -BI: Bitrasitive verb is one which requires two objects; one is direct object and another indirect object.

The transitive, intransitive, bitransiive verbs are further classified as finite and non finite forms. Just as nouns are marked for features like gender, number and case, verbs are also marked for certain features. The features marked are shown in table 3.4

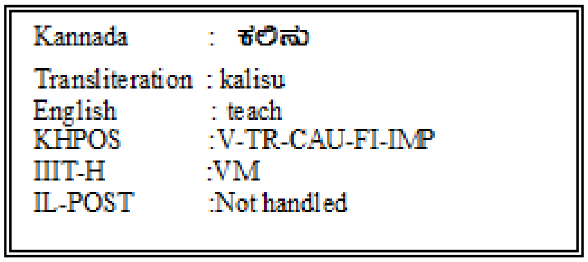
Table 3.4. Tags for Features Marked on Verb

S.No	Features Marked on Verb	Description	Tag
1	Type	Finite, non finite	FI, NF
2	Number	Singular, Plural	SL, PL
3	Person	First, Second, Third	P1, P2, P3
4	Tense	Present, Past, Future	PRES, PST, FUT
5	Aspect	Iterative, Durative, perfect	ITER,DUR,PERF
6	Auxiliaries	Modal, Aspectual	AUX
7	Voice	Active, Passive	ACT,PASS
8	Negation	illa,	NEG

3.3.9.2 Voice Modifier

The function of voice modifiers is to add voice distinction. Causative suffix ಇಸು/isu is used to mark voice modifier. In Kannada there is a causative suffix /isu/ that is added to the verb stems to make causative roots from non causative roots. Say the verb learn***“kali”* ಕಲಿ (*kali)* +/ಇಸು(*isu)* /➔ಕಲಿಸು*(kalisu)*** becomes ’teach’. Aspect markers can be attached after causativation. Which gives a meaning of passive? However causative suffix is optional. Care should be taken to handle the verbs which are already in causative form at the root level say**ಪ್ರಶ್ನಿಸು(*pranisu)***, causative suffix /isu/ should not be added to already causative roots, otherwise invalid forms like ***panisu+isu*➔*pranisisu*ಪ್ರಶ್ನಿಸಿಸು** a invalid form will be generated. We have taken this precaution in our morphological generator system. Example is shown in box 3.7

.



Kannada	: ಕಲಿಸು
Transliteration	: kalisu
English	: teach
KHPOS	:V-TR-CAU-FI-IMP
IIIT-H	:VM
IL-POST	:Not handled

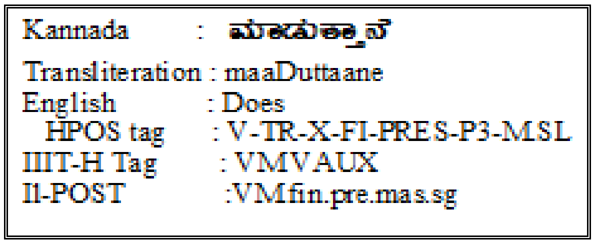
Box 3.7. Verb tag for Causative Form

3.3.9.3 Finite- (FI) Verbs

Verbs are further classified as non finite and finite forms. A finite verb is one that can stand as main verb of the sentence and always occur before full stop. A Finite verb has the following characteristics. Finite form is formed by adding appropriate ***tense suffix + personal suffix (gender+ number +person)*** to the verb stem. General structure of finite verb tag is.

i. V-Type (TR/IN/BI)-(Optional Voice) CAU-REF-FI-Tense (PRES/PST/FUT)-Aspect (PERF/ITER/DUR)-PNG+CLITIC.
ii. V-Type (TR/IN/BI)-(Optional Voice) CAU-REF-FI-DEF
iii. V-Type (TR/IN/BI)-(Optional Voice)CAU-REF-FI-(IMP/HORT/CNTG)

Consider the examples like maaDuttaane (he does), maaDidanu (he did), and maaDuvanu (he will do) are examples for finite forms of the verb do in English. If the verb stem consists of only one root then it is called simple stem. Similarly if the verb stem consists of root +voice modifier (causative) or modal auxiliaries it is called a compound stem. Finite verbs are classified into simple, complex and compound verbs based on their stem formation. Box 3.8 gives an example of finite form.



Kannada	: ಮಾಡುತ್ತಾನೆ
Transliteration	: maaDuttaane
English	: Does
HPOS tag	: V-TR-X-FI-PRES-P3-MSL
IIIT-H Tag	: VMVAUX
Il-POST	:VMfin.pre.mas.sg

Box 3.8. Tag for Finite Verb

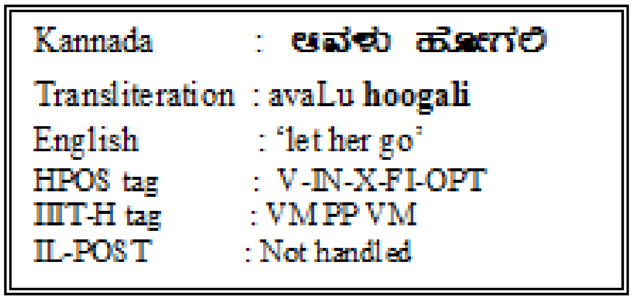
Kannada verbs exhibit a number of finite forms that express commands like singular *imperative and polite imperative, plural imperative, negative imperative* in second person singular, and *optative, hortative* are other finite forms shown in table 3.5.

Table 3.5. Sample Finite verbs

Finite verb	Tag	Suffix	Example
-Imperative -Polite imperative -Negative Imperative -Polite Negative Imperative	IMP	-command -request -command	maaDu (do) maaDi (please do) hooga beeDa/kuuDadu (don't go) hooga beeDiri/kkuDadu
Optative	OPT	Ali	avanu maaDali(let him do)
Hortative	HORT	ooNa	naavu tinnoNa (let us eat)
Contingent	CGNT	Yaanu	avanu maaDiyaanu (he might do)

3.3.9.4 Imperative-IMP: The Kannada imperatives express politeness in plural form by adding suffix /iri/ say ***nooDu+iru+i+➔nooDiri*** (please see) as stated in (Upadhyaya and Krishnamurthy 1972: 151) but to address female and male ***/ee/oo/*** are added say ***nooDee or nooDoo.***

3.3.9.5 Optative-OPT: Optative is a kind of imperative form in Kannada which is used with first and third persons. It is formed by adding /i/ to the infinitive form which refers to 'let' in English. Say ***hoogu + alu +i➔ hoogali as shown in box 3.9.***



Kannada	: ಅವಳು ಹೋಗಲಿ
Transliteration	: avaLu hoogali
English	: 'let her go'
HPOS tag	: V-IN-X-FI-OPT
IIIT-H tag	: VM PP VM
IL-POST	: Not handled

Box 3.9. Verb Tag for Optative

3.3.9.6 Hortative-HORT: Hortative is another kind of imperative (Biligiri 1959:81). It is formed by adding suffix /*ooNa*/to the verb stem as against the infinitive, as in thecase of other modals. It refers to shall we? of English. hoogu+*ooNa*➔hoogooNa, as shown in box 3.10.

naavu hoogoNa "Let us go"
HPOS Tag: V-IN-X-FI-HORT
IIIT-H Tag: VM, VAUX
IL-POST: Not Handled

Box 3.10. Verb Tag for Hortative

3.3.9.7 Contingent-CNTG: *The contingent feature is special to Kannada. This form does not exist in other south Indian languages like Tamil or Telugu.* Verb forms like ***maaDiyaanu* 🡐 *maaDu+i+y+aanu*** (he might have done) refers to contingent. *This feature is not handled in (Baskarann, 2008). This form is formed by adding suffix* **ಆನು** ***"yaanu"*** *to the past verbal participle form.*

3.3.9.8 Negative contingent: The modal **ಆರ್**"*aar*" with person, number, gender (PNG) markers, is attached to the infinitive form: the meaning refers to 'might not' in the sense of lack of ability. Say ***maaDu+alu+aar+Lu➔maaDalaaraLu.***

3.3.9.9 Defective Verbs –DEF. Kannada has a few verbs that do not behave morphologically and syntactically like other main verbs.Suchverbs are called as "defective" verbs by some grammarians: they lack many of the forms that the regular verbs generally have. They are called *"impersonal construction"* by (Upadhyaya and Krishnamurthy, 1972:38), and as Dative-stative verbs (Harold Schiffman, 1979:75) because semantically they are stative describing states rather than actions. These verbs usually have "subjects" in dative case. The modal beeku, "want" is also used as a defective verb when used without other lexical verbs. Defective verbs are shown in table 3.6. Defective verbs are not handled in (Baskaran, 2008).

Table 3.6. Showing Defective Verbs

Defective Verb	Meaning
Saakuu	Enough

Gottu	Know
ishTa	To one’s liking
Saaladu	Not sufficient
Illa	Exists not
Alla	No(not logical)
Sari	Okay
Tappu	Wrong

3.3.9.10 Non Finite –NF Verbs.

A non finite form is one that cannot stand as main verb of the sentence and rarely occurs before the full stop. The structure of a non finite verb is: verb root+optional causative+aspect (perfect / durative/iterative) +any nonfinite form. A Non-finite verb has the following characteristics:

i. It doesn’t carry gender, number and person in agreement with the grammatical subject of the sentence.

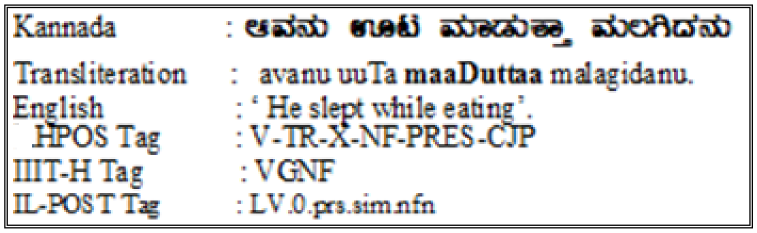
Conditional, concessive, participles, infinitives are different types of nonfinite forms.

3.3.9.10.1.1 Conditional form-COND: The conditional form in Kannada is formed by adding the suffix ಅರೆ/are/ to the past stem of the main verb, tense and aspect distinctions being made to this construction. Say, verb forms like maaDu +/id/ +/are/➔ maaDidare. This form gives two senses (If they do or if done) both senses are captured in our morphological analyzer/generator system.

3.3.9.10.2 Concessive form-CONC: The concessive form in Kannada is formed by adding the suffix ಅರೂ/aruu/ to the past stem of the main verb. Tense aspect distinctions can be made to this construction. Say verb forms like maaDu+/id/ +/aruu/➔ maaDidaruu. ‘though done’. Concessive Forms are not handled in IL-POST.

3.3.9.10.3 Participles: Participles are non finite verb forms that function verbally or adjectivally or have some special syntactic function in the sentence. Participles have tense, and aspect inflections. Participles can be classified as conjunctive and relative participles.

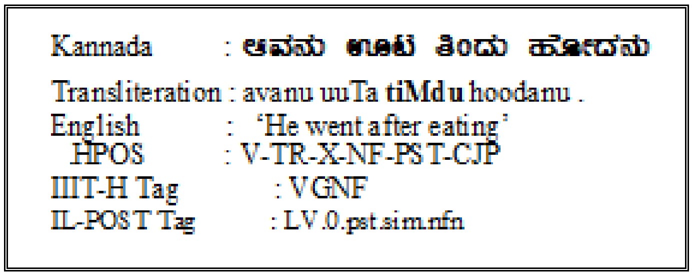
3.3.9.10.3.1 Present verbal participle is formed by adding suffix 'aa' to the present tense suffix "utt" say maaDu+utt+aa➔maaDuttaa. Present verbal participle is usually followed by finite verb say "maaDuttaa oDidanu/ ran while doing" indicates that the actions of the participle verb and main verb are simultaneous. Consider an example shown in below box 3.11.



Kannada	: ಅವನು ಊಟ ಮಾಡುತ್ತಾ ಮಲಗಿದನು
Transliteration	: avanu uuTa **maaDuttaa** malagidanu.
English	: ' He slept while eating '.
HPOS Tag	: V-TR-X-NF-PRES-CJP
IIIT-H Tag	: VGNF
IL-POST Tag	: LV.0.prs.sim.nfn

Box 3.11. Present Verbal Participle

3.3.9.10.3.2 Past verbal participle is formed by adding'i/u' to the verb root, say tinnu+i➔ tiMdu + biDu +i ➔ tiMdubiTTu (having finished eating). In Kannada we can attach a set of aspect auxiliaries shown in table 3.7, to the past verbal participle to coin new verb forms. The formation of new verbs is a productive rule. Consider an example in box 3.12.

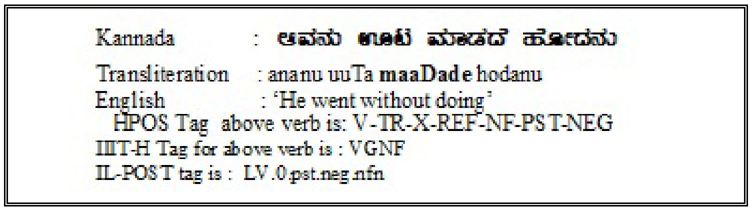


Kannada	: ಅವನು ಊಟ ತಿಂದು ಹೋದನು
Transliteration	: avanu uuTa **tiMdu** hoodanu .
English	: 'He went after eating'
HPOS	: V-TR-X-NF-PST-CJP
IIIT-H Tag	: VGNF
IL-POST Tag	: LV.0.pst.sim.nfn

Box 3.12. Past Verbal Participle

3.3.9.10.3.3 Negative verbal participle – NEG

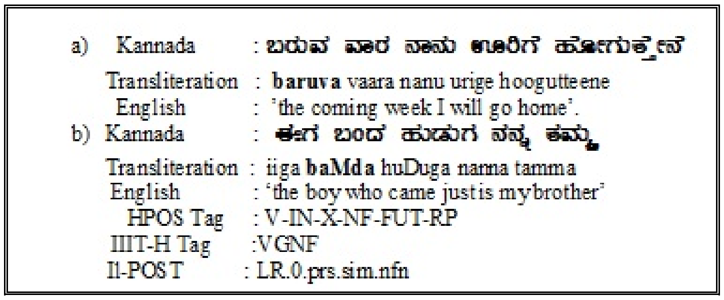
Negative verbal participles are not inflected for tense markers, the present and past verbal participles uses a common suffix (ade) to express the notion 'not doing' or 'not having done'. Say maaDu+ade➔maaDade (without doing). Shown in box 3.13.


Kannada : ಅವನು ಊಟ ಮಾಡದೆ ಹೋದನು
Transliteration : ananu uuTa **maaDade** hodanu
English : 'He went without doing'
HPOS Tag above verb is: V-TR-X-REF-NF-PST-NEG
IIIT-H Tag for above verb is : VGNF
IL-POST tag is : LV.0.pst.neg.nfn

Box 3.13. Verb Tag for Negative Verbal Participle

3.3.9.10.3.4 Relative Participle - RP

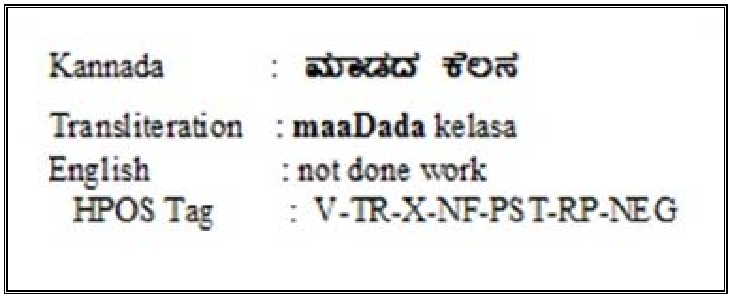
The relative participles are derived from verbs and these derivations function similar to relative clauses in English. They not only modify the noun that follows but also take part in the formation of noun phrases. Though relative participles behave as adjectives but they possess verb properties since they are derived from verbs. So it is necessary to keep them under a separate participle category. The present adjectival participles are formed by adding suffix 'a' to future/past stem say baru+uv+a➔ baruva (coming), baru+id+a➔baMda (who came). Consider an example in box 3.14.


a) Kannada : ಬರುವ ವಾರ ನಾನು ಊರಿಗೆ ಹೋಗುತ್ತೇನೆ
Transliteration : **baruva** vaara nanu urige hoogutteene
English : 'the coming week I will go home'.
b) Kannada : ಈಗ ಬಂದ ಹುಡುಗ ನನ್ನ ತಮ್ಮ
Transliteration : iiga **baMda** huDuga nanna tamma
English : 'the boy who came just is my brother'
HPOS Tag : V-IN-X-NF-FUT-RP
IIIT-H Tag :VGNF
Il-POST : LR.0.prs.sim.nfn

Box 3.14. Verb Tag for Relative Verbal Participle

3.3.9.10.3.5 Negative relative Participle - NEG-RP

The negative adjectival participle is formed by adding suffix 'ada' to the verb stem. These forms are like maaDu+ada➔maaDada (not done) shown in box 3.15.


Kannada : ಮಾಡದ ಕೆಲಸ
Transliteration : **maaDada** kelasa
English : not done work
HPOS Tag : V-TR-X-NF-PST-RP-NEG

Box 3. 15. Verb Tag for Negative Relative Verbal Participle

3.3.9.11 Infinitive –INF

Infinitive is another nonfinite form. The infinitive form of a verb will appear in the dictionary, this is true for English but not for Kannada. In the case of Kannada we treat the imperative form of a verb as basic form. The infinitive form of a verb is usually preceded by to, (to run, to dance, to think in English) followed by the simple form of the verb. In Kannada suffix /alu/ is used as infinitive marker attached after the verb. Say maaDu+alu➔maaDalu (to do).

3.3.9.11 Aspect auxiliaries

Formation of new verbs by adding a set of polar verbs to the lexical verbs generates many verbs in Kannada. This is a productive rule in Kannada. *In Kannada a set of verbs may be added to the past verbal participle form to give certain semantic nuances to the meaning of the sentence.* These set of verbs are called as aspect markers. These are very similar to main verbs in their morphology and syntax. *But semantically they do not express the same lexical meaning when used as main verbs.* The aspectual *biDu 'completive' does not mean the same as the main verb biDu "leave". This feature is not captured in (Baskaran et. al, 2008).* Handling of aspect auxiliaries is an important factor because using this feature infinite number of verb stems can be generated. The list of aspect auxiliaries is shown in table 3.7. We have captured this aspect in our morphological analyzer/generator.

Table 3.7. Showing the List of Aspect Auxiliaries

S.no	**Aspect Marker**	**Aspect Meaning**	**Lexical Meaning**
1	biDu	Completion	Leave
Consider an example Kannada:**ಅವನು ಹಣ್ಣುತಿಂದುಬಿಟ್ಟನು** ... (1) Transliteration :avanu haNNu tiMdubiTTanu English: He ate up the fruit			
2	Hoogu	Completion	Go
Consider an example Kannada: **ನೀನು ಉಟ ಮಾಡಿಹೋಗು**_ (2)			

Transliteration: niinu uuTa maaDihooguEnglish :You take meals and go			
3	aaDu	Continuity	Play
Consider an example Kannada: ಅವರು ಕುಣಿದಾಡಿದರು_ (3) Transliteraion: avaru kuNidaaDidaru English: They danced and played			
4	koDu	Benefactive	Give
Consider an example Kannada:ನೀನು ಆ ಕೆಲಸವನ್ನು ಮಾಡಿಕೊಡು_ (4) Transliteration: niinu aa kelasavannu maaDikoDu English: you that work after doing give			
5	nooDu	Attemptive	See
Consider an example Kannada:ಮನೆ ಕಟ್ಟಿನೋಡು_ (5) Transliteration: mane kaTTinooDu English: build the house and see			
6	Haaku	Exhaustive	Put
Consider an example Kannada:ಅನ್ನ ಬೇಯಿಸಿಹಾಕು_ (6) Transliteration: anna beeyisihaaku English: cook the rice			
7	koLlu	Reflexive	Purchase
Consider an example Kannada: ಅವನು ಬಟ್ಟೆ ಹಾಕಿಕೊಂಡನು_ (7) Transliteration: avanu baTTe haakikoMDanu English: He dressed up			
8	Iru	Perfective	Be
Consider an example Kannada: ನೀನು ಸುಮ್ಮನೆ ಕುಳಿತಿರು_ (8) Transliteration: niinu summane kuLitiru English: you keep sitting silently			

3.3.9.13 Modal auxiliaries

Kannada has a number of modal auxiliaries that are usually attached to the */al/* form of infinitive forms. Different modal auxiliaries *give the notion of could, can, may, might, should, capable, ability, compulsiveness, completion, prohibitive, permissive and opportunity and their negatives.* Kannada modals are asymmetrical in the negative form that is the negative forms do not exactly parallel the affirmative. *This feature is currently not handled in (Baskran et.al, 2008).* This feature is also useful in the generation of many derived verb roots. We have handled this feature; this is also one of the factors that increased the tagging rate of Kannada HPOS tag set. The different models are listed in table 3.8.

Table 3.8. Table showing modal auxiliaries

Modal auxiliary	Meaning	Example
Beeku	MUST (Want)	huuvu beeku (want a flower)
kuuDadu	PROH(Should not)	niivu hoogakuuDadu (you should not go)
beeDa	NEG(IMP)	pustaka beeDa (I do not want the book)
Bahudu	PERM(May)	niivu barabahudu (you may come)
aara	NCAP(might not)	avanu tinnalaranu (he cannot eat)
Balla	CAP(capable)	avanu maaDaballanu (he can do)
paDu	PASS(Passive voice)	haNNu tinnalapaDuttade (Fruit will be eaten)

- **Modal beeku**

The modal beeku refers to 'want' in English; the auxiliary modal beeku is attached to the infinitive form and gives the meaning *wants something*. When clitics get added to it, the meaning changes to the emphatic of beeku, say, referring to 'must' in English. Modal 'beeku' is not inflected for tense or PNG markers. However it can occur with other aspect auxiliary like aagu 'become' and iru 'be' to give other senses. The modal beeku appears in dative stative construction, with the subject marked in dative case as shown in box 3.16.

nanage *odalu* pustaka beeku (I need book to read)
HPOS Tag: V-IN-X-NF-INF
IIIT-H Tag: VNF
IL-POST Tag: VM.0.inf.nfn

Box 3.16. Verb Tag for Infinitive

- **Modal beeDa**

  'beeDa' is negative of 'beeku'. The negative beeDa refers to 'do not' in English, as in verb forms like /hoogabeeDa/ donot go.

- **Modal 'kuuDadu**

  'kuuDadu' is negative imperative. It is attached to the infinitive form. It corresponds to 'should not' of English, as in verb forms like /barakuuDadu/ 'should not come'.

- **Modal bahudu**

  The auxiliary 'bahudu' is attached to the infinitive and has the meaning of 'may'. The permissive sense, as in verb forms like maaDu +/alu +/ bahudu/➔ maaDalubahudu' can be done'.

- **Modal 'baaradu**

  'Modal 'baaradu' is negative imperative. The negative of bahudu is baaradu, it is attached to the infinitive. It gives the sense of 'cannot' and 'should not', as in verb forms like (maaDu +/alu/+ /baaradu/➔ maaDalubaaradu 'should not do'.

3.4. Abbreviations Used in Hierarchical Tagset

We have developed tag set using 170 tag elements. The abbreviations used in the design of the tagset are shown in table 3.9.

Table 3.9. The List of abbreviations Used in Hierarchical Tags

Term	Abbreviation	Tag	Abbreviation	Tag	Abbreviation
N	Noun	PRO	Pronoun	RIB	Right bracket
COM	Common	PER	Personal	ANG	Angular
COU	Countable	INTG	Interrogative	SQR	Square
UNC	Uncountable	REFL	Reflexive	FLR	Flower
SL	Singular	INDF	Indefinite	PAR	Parenthesis
PL	Plural	EMP	Emphatic	SNG	Single quote
NOM	Nominative	INCL	Inclusive	DBL	Double quote
GEN	Genitive	ADV	Adverb	SEP	Separating mark
DAT	Dative	MAN	Manner	FUS	Full stop
ACC	Accusative	ADJ	Adjective	HYP	Hyphen
ABL	Ablative	DEM	Demonstrative	EXC	Exclamatory
PUR	Purposive	ORD	Ordinal	QUE	Question mark
COMP	Comparative	CARD	Cardinal	COMA	Comma
LOC	Locative	ABS	Absolute	V	Verb
LOCD	Locative dative	QNTF	Quantifier	PUN	Punctuation
PLA	Place	NEG	Negative	NF	Non finite
TIM	Time	QW	Question	CJP	Conjunctive participle
CARD	Cardinal	COND	Conditional	RP	Relative Participle
HUM	Human	COOR	Coordinate	NEG	Negative
NHU	Non-human	SUB	Subordinate	PST	Past
PP	Postposition	INTF	Intensifier	PRES	Present
INTJ	Interjection	LEB	Left bracket	FUT	Future
CONJ	Conjunction	CONC	Concessive	DUR	Durative
FI	Finite	TR	Transitive	INTR	Intransitive
BI	Bi-transitive	CAU	Causative	IMP	Imperative
HORT	Hortative	PIMP	Polite imperative	NPO	Negative Potential
PASS	Passive	PROH	Prohibitive	CAP	Capablative
NPER	Non permissive	PERM	Permissive	NCAP	Non Capablative

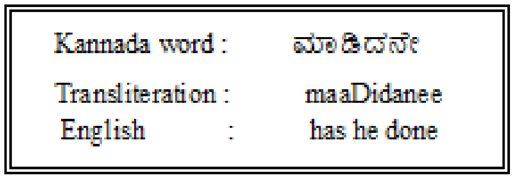
Term	Abbreviation	Tag	Abbreviation	Tag	Abbreviation
PROB	Probable	OBL	Oblique	P1/P2/P3	First/second/Third Person
M/F/N	Masculine/feminine/neuter	PTN	Proto noun	AFF	Affirmative
SIM	Similarative	SOC	Sociative	DIST	Distance
PROX	Proximate	INF	Infinitive	MA	Modal auxiliaries
OTH	Others	DUP	Duplicate	DFLT	Default
COL	Colon	SEC	Semicolon		

3.5 Improvement of HPOS Tagset over Microsoft IL-POST Framework

(Baskaran et al., 2008) have proposed Common Frame work guidelines for Indian languages using EAGLES as the basis model. It is referred to as IL-POST. Our design of tag set is developed keeping parsing in mind. So it is felt that semantic information is also required to some extent, for example, what kind of argument the verb takes, and marking of transitive, intransitive, bitransitive information is required in higher application like Machine translation, Anaphora resolution application, parsing etc. IL-POST tagset does not capture such information. Another major issue is, in Kannada *the interrogative sentences are formed by using clitics, this information is also missing in IL-POST framework. Formation of conjunctive verbs and compounding are missing in IL-POST tag set.* To start with, we focus on clitics first, 4 types of clitics are identified.

1. ***Clitics*** information is not handled in (Baskran et al., 2008) guidelines. Clitics provide lot of information for higher level analysis: for example during parsing. Handling clitics is focused in our Kannada HPOS tagset. Four clitics are identified viz. Emphatic clitic, Interrogative clitic, Inclusive clitic and Infinitive clitic. However one cannot ignore the occurrence of clitic in multilevel, i.e. clitic emphatic is followed by interrogative clitic.*This was an open issue in Baskaran et al. tagset, which we have, overcome here.*

   Clitics are the particles that can be added to any constituent of a sentence except adjective, with different semantic effect. Depending on the constituent, which have the whole sentence as their hosts? In linguistics, a clitic is a grammatically independent and phonologically dependent morpheme. It is pronounced like an affix. Clitics are required to indicate elongate emotions. *Clitics cannot be added to adjectives and finite verbs (Harold Schiffman, 1979:131), that cliticscannot be added to adjective is accepted but* ***we argue that "clitics are not added to finite verbs****". This argument is not correct. Since we have many examples in Kannada which show* ***we can add clitics to finite verbs*** *as shown in below examples in Box 3.17*



Kannada word	:	ಮಾಡಿದನೇ
Transliteration	:	maaDidanee
English	:	has he done

Box 3.17. Finite Verb With Emphatic Clitic

Here the word maaDidanee = maaDu + id+ anu+ ee (Emphatic clitic). The word form maaDidanee is formed by adding the emphatic clitic to the finite form maaDidanu (he has done) + ee. This is a valid verb form giving interrogative meaning

i. ***ee-emphatic clitic*****:** The emphatic clitic is used to indicate emphasis. Ramanujan (1963:70) calls it an 'exclusive' particle. The addition of 'ee' to host emphasizes the character of that host to the exclusion of other things of presupposition.

ii. ***oo-indefinite clitic*****:** When clitic 'oo' is added to a constituent, it indicates that the speaker has some doubt or uncertainty about the host of a clitic. Also it may indicate that the speaker has just realized that some presumption he had about the situation has now been confirmed. The speaker is asking for information about the truth, or confirmation of his belief about something, in such case'oo' clitic can be used. When two or more same type constituents have 'oo' added to them, the meaning is 'either' or 'or' like avanoo, nanoo barutteeve . Here 'oo' is acting as coordinating conjunction.

a) ರಾಮನು ಶಾಲೆಗೆ ಹೋದನೊ b) ಸೀತೆಯು ಬಂದಳಾ

(Did raama go?)	(Did sita come)
(raamanu hoodanoo)	(siiteyu baMdaLaa)

Here the indefinite clitic 'oo' is gives interrogative meaning. But this sense of interrogation is different from another interrogative clitic 'aa', when the speaker is asking for information about the truth, or confirmation of his belief about something, the 'oo' clitic can be used. For example consider the sentence for machine translation (MT) by IL-POST and Our Tag. By IL-POST tagging the first 2 words remained untagged due to clitics. This hinders the MT application, since untagged words are not translated. But by using HPOS tag set, all words are tagged.HPOS tag is useful for machine translation application.

d) ಅವನೋ ನಾನೋ ಬರುತ್ತೇವೆ (Either he or I will come)

< ? ><?><VM-0.pl.prs.fin> :- IL-POST Tags, no tag for cliticized inflection.

<PRO-PER-P3.M.SL.NOM-EMP><PRO-PER-P2.SL.NOM-EMP><V-IN-PRES.P1.PL>

Here 'oo' is acting as coordinating conjunction. PRO-PER-P2.SL.NOM-EMP

iii. ***uu-inclisive clitic:*** When added to the words, gives the meaning of inclusiveness, it is also used to conjoin two similar constituents with the meaning 'and'. For example consider the sentence avanuu baMdanu, "he also came". When 'uu' is added to question words like elli (where) it becomes elliyuu, the original interrogative meaning of the word "elli" (where) is lost.

iv. ***aa- Interrogative clitic:*** When this clitic is added, it gives the meaning of indefiniteness like, avanaa? (is he)?.

2. Semantic information like transitive, intransitive are marked on verbs in our work. But Baskran et al. have not handled this presently, *this information is important in finding what type of argument the verb takes and this information is useful in resolving ambiguity, when there are multiple analyses produced by morphological analyzer.*

3. IL-POST has no finer distinctions in adjectives. However this distinction is necessary because *the order of occurrence of adjectives provides useful information in a noun phrase grouping*, like for instance which adjective, follows which kind of another adjective; *it is observed that all true adjectives, quantifiers and ordinals cannot be kept in the same order.* Chunking rules are Determined based on occurrence of constituents in a phrase, for example consider the sentence.

   a)ಸುಂದರ ಇನ್ನೊಬ್ಬ ಹುಡುಗ ಇದ್ದಾನೆ ... is an ungrammatical sentence.

   suMdara innobba huDuga iddaane ...Transliteration

   (Handsome another boy)...English Equivalent

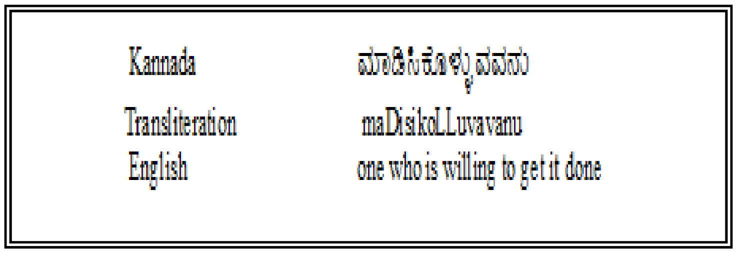
   b)ಇನ್ನೊಬ್ಬ ಸುಂದರ ಹುಡುಗ ಇದ್ದಾನೆ...correct grammaticalsentence.

   *innobba suMdara hyDuga iddaane (there is another beautiful boy) is*the grammatically correct form. Order of occurrence is the true adjective precedes the other adjectives therefore classification of adjectives is an important factor which cannot be ignored.

4. We have decomposed common noun as countable and uncountable. This distinction is necessary because uncountable nouns have only case inflection but not number i.e. plural inflection. This information is useful in word form generation, otherwise morphological generation systems keep on generating ungrammatical forms like below. This information is missing in Microsoft guidelines. Consider the word as given in Example.
   a) ನೀರುಗಳು niirugaLu (water) in plural not correct grammatically.

5. It is ideal to have participles as a subcategory under verbs non finite, since in Dravidian tradition the verb non finite form takes the form of participles or infinitive. Since top level category should be semantically motivated rather than motivated by the grammatical properties, participles are grouped not as top level category but as verb non finite forms in our work, unlike as top level category in IL-POST. This information is captured through derivation morphology in our case.

6. In Microsoft framework guidelines, nouns derived from relative particles and nouns derived from adjectives are placed under a single category as Nominal under Participle. But this aspect is taken up in derivation morphology in our work. The derivation process is described in the below example, shows that verbal noun is derived as RP (relative participle) later, third person pronoun suffixes are added to it to get noun derivations. This is a productive process in Kannada. The relative participles (RP) behaves as adjective but still carries features of verb, like perfect, durative aspect features and other features like causative or reflexive, which could not be captured properly if it is treated simply as Nominal. There has to be some indication to show the noun is derived from a relative participle. Consider an example in box 3.18.



Kannada	ಮಾಡಿಸಿಕೊಳ್ಳುವವನು
Transliteration	maDisikoLLuvavanu
English	one who is willing to get it done

Box 3.18. Tag for Noun Derived From Relative Participle

The HPOS tag is (V-TR-CAU-CJP-REF-FUT-RP-PRON.avanu). Verb-transitive-causative-reflexive-future- avanu in singular form, relative participle-pronoun third person. Though relative participle behaves as adjectives, they are derived from verbs, they have features of verbs. So they are kept separately under a different category in our design. The structure of relative participles is root+voice modifiers+aspect+ modal auxiliaries+ tense suffix.

7. *Another issue is that nouns derived from adjectives, have no tense, aspect information, they are derived from adding the third person pronoun suffix.* For example consider the word below. These word forms function as nouns but are derived from adjectives.

a) cikkavanu (younger) is tagged as (ADJ-ABS-PRON. avanu), cikka is an adjective.

8. ***Another open issue proposed in Microsoft guidelines is that it is hard to distinguish between adjectival participles and verbal nouns for Bangla.*** But however this problem can be solved in Kannada. In Kannada there is a productive rule for deriving nouns from adjectives by adding third person pronoun suffix ‘avanu’ as shown in the above previous example ‘cikkavanu’. Relative participles obey this rule but verbal nouns do not satisfy this rule so one can distinguish between adjective participles and verbal nouns easily. Consider another example in box 3.19.

Kannada	:	ಒಳ್ಳೆಯವನು
Transliteration	:	oLLeyavanu
English	:	good person

Box 3.19. Tag for Noun Derived from Adjective

This noun derived from the adjective oLLeya (good) by conjugating it with ‘avanu’ (he) which is a personal pronoun. Relative participles obey this rule but verbal nouns do not satisfy this rule so one can distinguish between adjective participles and verbal nouns.

9. Relative participles and conjunctive participles are kept under verb non finite forms in Microsoft guidelines work. In our work Even though relative participles act as adjectives they are not placed under adjectives since these can take the negative suffix as shown in below example. *Kannada does not have negative adjectives as in Dravidian Languages. Hence these should be treated separately.* Consider a word like baarada (not coming one), here baarada is a negative relative participle acting as an adjective Example shown in box 3.20.

Kannada	:	ಬಾರದ ಹುಡುಗ
Transliteration	:	baarada huDuga
English	:	Boy who did not come

Box 3.20. Negative Relative Participle

10. Demonstratives are placed as top level category in Baskaran et al. framework, relative pronouns and wh-words are considered as types in demonstratives,

which are also considered under pronouns as types. This information is redundant. In Kannada 'aa' (that), 'ii' (this) are treated as *demonstratives these don't have plural attributes unlike Baskaran et al proposal. Hence we have demonstratives kept under adjectives for 2 reasons.*

i) The difference between demonstrative adjectives and neuter demonstrative pronouns (that, this) is difficult for English speakers to negotiate because, this, that, which, what are used as pronouns and also as adjectives in English.

j) idu pustaka. (This is a book) h) ii huDuga (This boy).

In Kannada idu (this) is different from "ii". We have kept idu and adu like words under Personal pronouns. Words idu/adu have plural forms as ivu and avu. *As Kannada does not have plural adjectives.* Another thing is that demonstrative adjectives in Kannada are always used before nouns as noun modifiers. Hence the rule that a pronoun replaces a noun is violated. (This boy) and also it requires some noun immediately as its argument as in this example 'boy'. In our tagset design we have overcome these conflicts.

11. *In Baskaran et al. Framework ordinals (like oMdaneeya, first) are kept under Quantifiers, and cardinals (oMdu, one) are also kept under quantifiers this distinction is not correct,* because though ordinals and cardinals represent numbers, ***they play different roles in sentences, ordinals are not inflected for cases and cardinal are inflected for case and cardinals can be head of the noun phrase, while ordinals cannot, and ordinals are more like adjectives, they are used in deriving nouns like other adjectives.*** Considering these features, ordinals are kept under adjectives since they act as noun modifiers like adjectives in our HPOS Tag set.

12. Cardinals are placed under the noun category. Human and nonhuman are its subtypes, since they take case inflections hence are placed under nouns. Consider an example say *hattariMda guNisu* (multiply by ten, "hattariMda" is tagged as, N-CARD-NHU-ABL), noun cardinal nonhuman with ablative case inflection.

13. Particles are kept in top level category in Microsoft guidelines. Conjunction and interjections are placed in it as subtypes. But In Kannada Interjection and conjunction are kept separately as a top level category. Subordinating, coordinating conjunctions are treated as a sub category under Conjunction.

14. Relative pronouns do not exist for Kannada.

15. Reciprocals are placed under Pronoun in IL-POST, which is not correct. But reciprocals like 'obbobba' has a double functionality i.e. they are used as noun modifiers for example "obbobba vyakatiyannu nooDuve," (I will see each each person). In this example the *reciprocal is modifying noun but the general theory of language is a pronoun replaces a noun is an accepted rule but a pronoun modifying a noun is not accepted theory, hence in our tag set, such words are treated as adjectives and they are treated specially as* ***protonoun*** since they are further derived as nouns like obbobbanu (indicating masculine single person), obbobaLu (indicates feminine).

16. Aspect auxiliary verbs are not handled in Microsoft Guidelines. We have handled this since it is important information in the generation of compound verbs. A combination of verbal participle followed by a set of auxiliary verbs, then followed by aspect and tense inflections is also handled in our HPOS tag set. A set of verbs biDu, hogu, haaku, aaDu, nooDu, koDu, koLLu, baru, iDu, aagu are studied under aspect markers apart from their usage as main verbs. But when these verbs are used as aspect, they impart different meanings, for example. biDu (leave) when used as aspect in the compound verb tiMdu biDu (eat up), the original meaning of biDu is not meant here. IL-POST has not handled this property. We have handled Compound verbs like these.

17. In IL-POST guidelines, **there is no detailed distinction made for adverb also.** However this division is necessary for handling of onomatopoeic forms like sarasarane (fast fast), barabarane (quickly), which indicates emphasis is required. *The finer distinction is necessary since all types of adjectives or adverbs occur in a particular order. Information regarding which subtype*

follows which subtype is useful in chunking while identifying noun groups and verb groups.

We have resolved many unsolved open issues mentioned by the IL-POST tag set. And so can easily prove that our tagset is an exhaustive tag set for Kannada.

Table 3.10. Comparison of IL-POST and Our Kannada HPOS Tag

S.No	**HPOS for Kannada**				**IL-POST for Indian Language**			
	Top Level	**Sub category (Level 1)**	**Sub- sub category (Level 2)**	**Tag**	**Top Level**	**Sub cat gory (Level 1)**	**Sub sub catego ry (Level 2)**	**Tag**
1	**Noun**			**N**	**Yes**			**N**
		Common		**COM**		**Yes**	**Yes**	**C**
			Countable/ Uncountable	**COU/UNC**			**No**	**-**
		Proper			**Yes**			**P**
			Person	**PER**		**No**		**-**
			Location	**LOC**		**No**		**-**
			Organization	**ORG**		**No**		**-**
		Locative		**LOC**	**Yes, Merged**			**NST**
			Time	**TIM**		**No**		
			Place	**PLA**		**No**		
		Cardinals		**CARD**	**Numerals**			
			Human	**HUM**		**Cardinals**		**NUMC**
			Non human	**NHUM**		**Ordinals**		**NUMO**

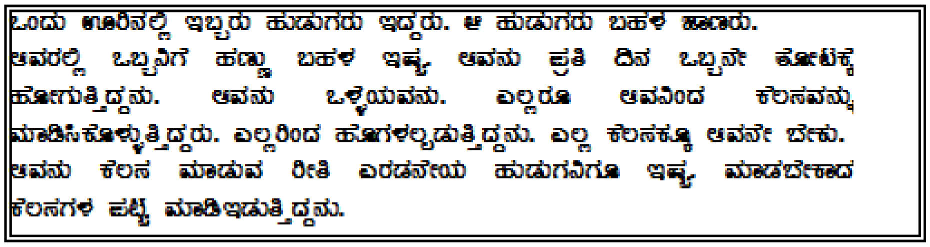
2	**Pronoun**			**PRP**	**Yes**			**PPR**
			Personal			**Yes**		
				PROX				
				DIST				
			Reflexive			**Yes**		**PRF**
			Interrogative			**Yes**		**PWH**
			Kept under cardinals as human		←	**Reciprocal**		**PWC**
3	**Adjective**			**ADJ**	**Nominal**	**Adjective**		**JJ**
						Quantifier		**JQ**
		Demonstrative		**DEM**		**No**		
		Quantifiers		**QNTF**		**No**		
		Ordinals		**ORD**		**No**		
		Absolute		**ABS**		**No**		
4	**Adverb**			**ADV**				
		Time		**TIM**		**No**		
		Place		**PLA**		**No**		
		Reduplication		**RDP**	**No division**	**No**		
		Question		**QW**	**In adverbs**	**No**		
		Intensifier		**INTF**		**No**		
		Absolute		**ABS**		**No**		
		Conjunction		**CONJ**		**No**		
5	**Interjection**	-		**INTJ**		**Yes**		**IN**
6	**Conjunctuation**	Coordinating		**COOR**		**- -**		
		Subordination		**SUB**				

7	Punctuation				**PUN**				**PU**
		<			**PUN-LBR-ANG**		**No**		
		{			**PUN-LBR-FLR**		**No**		
		(			**PUN-LBR-PAR**		**No**		
		[			**PUN-LBR- SQR**		**No**		
		>			**PUN-RBR-ANG**		**No**		
		}			**PUN-RBR-FLR**		**No**		
		]			**PUN-RBR-SQR**		**No**		
		:			**PUN-SEP-COL**		**No**		
		,			**PUN-SEP-COMA**		**No**		
		!			**PUN-SEP-EXC**		**No**		
		?			**PUN-SEP-QUE**		**No**		
		;			**PUN-SEP-SEMC**		**No**		
		-			**PUN-SEP-HYP**		**No**		
		“			**PUN-QUO-DBL**		**No**		
8	**Verbs Details shown in figure 3.3**		**VM**		**V**	**Yes**			**V**
		TR		**FI**			**Finite**		
		IN		**INF**			**Non finite**		**FIN**
		BI		**NF**					**NFN**
							Infinitive		**INF**
9	**Postposition**				**PP**				
		Genitive			**PP-GEN**		**No**		
		Ablative			**PP-ABL**		**No**		
		Dative			**PP-DAT**		**No**		
		Accusative			**PP-ACC**		**No**		
		Comparative			**PP-**		**No**		

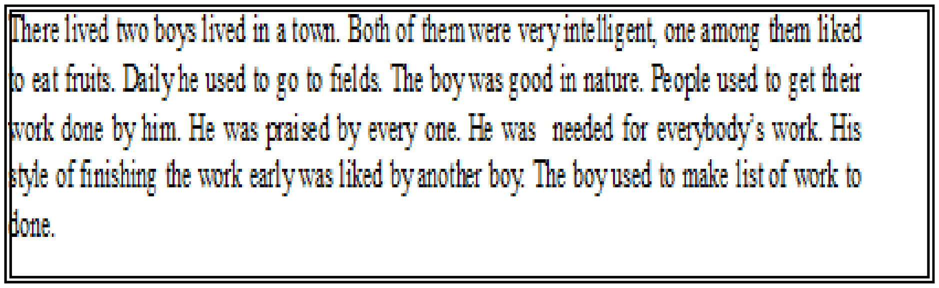
			COMP			
		Purposive	**PP-PUR**		**No**	
		Similarities	**PP-SIM**		**No**	
		Associative	**PP-SOC**		**No**	
		Others	**PP-OTH**		**No**	
10	**UNK**		**RD**	**Yes**		**RD**

3.6. Results

To check the tagging efficiencies we collected samples text from the Kannada website, Kannada yahoo.com, sample from famous Kannada daily news paper Prajaavaani and synthetic sample generated manually. The samples are transliterated by using the Unicode to roman converter program ur.pl. Samples are tagged manually by using different tagsets like IIIT-Hyderabad tagset and Microsoft IL-POS tagset and HPOS tagset. The samples of the original text, transliterated text and corresponding English versions are shown in below boxes. The synthetic sample is shown in box 3.21, the transliterated version is shown in box 3.22 and English version is shown in box 3.23.


ಒಂದು ಊರಿನಲ್ಲಿ ಇಬ್ಬರು ಹುಡುಗರು ಇದ್ದರು. ಆ ಹುಡುಗರು ಬಹಳ ಜಾಣರು.
ಅವರಲ್ಲಿ ಒಬ್ಬನಿಗೆ ಹಣ್ಣು ಬಹಳ ಇಷ್ಟ. ಅವನು ಪ್ರತಿ ದಿನ ಒಬ್ಬನೇ ತೋಟಕ್ಕೆ
ಹೋಗುತ್ತಿದ್ದನು. ಅವನು ಒಳ್ಳೆಯವನು. ಎಲ್ಲರೂ ಅವನಿಂದ ಕೆಲಸವನ್ನು
ಮಾಡಿಸಿಕೊಳ್ಳುತ್ತಿದ್ದರು. ಎಲ್ಲರಿಂದ ಹೊಗಳಲ್ಪಡುತ್ತಿದ್ದನು. ಎಲ್ಲ ಕೆಲಸಕ್ಕೂ ಅವನೇ ಬೇಕು.
ಅವನು ಕೆಲಸ ಮಾಡುವ ರೀತಿ ಎರಡನೇಯ ಹುಡುಗನಿಗೂ ಇಷ್ಟ. ಮಾಡಬೇಕಾದ
ಕೆಲಸಗಳ ಪಟ್ಟಿ ಮಾಡಿಇಡುತ್ತಿದ್ದನು.

Box 3.21.Generated Sample Synthetic Kannada Input Text


There lived two boys lived in a town. Both of them were very intelligent, one among them liked to eat fruits. Daily he used to go to fields. The boy was good in nature. People used to get their work done by him. He was praised by every one. He was needed for everybody's work. His style of finishing the work early was liked by another boy. The boy used to make list of work to done.

Box 3.22. **English Translation of Kannada Text given in Box 3.21**

oMdu urinalli ibbaru huDugaru iddaru. Aa huDugaru bahaLa jaaNaru. avaralli obbanige haNNu bahaLa ishTa. avanu pratidina obbanee toTakke hoguttiddanu. avanu oLLeyavanu. ellaru avaniMda kelasavannu maaDisikoLLu ttiddaru. ellariMda hogaLalpaDuttiddanu Ell kelasakkuu avanee beeku. avanu kelasa maaDuva riiti eraDaneeya huDuganiguu ishTa. maDabeekaada kelasagaLa paTTi maaDiiDuttiddanu.

Box 3.23 Transliteration of Kannada Text in Box 3.21.

i) <oMdu > < JQ.0.sg.dir.n.crd> < uurinalli> <N.0.sg.loc> < ibbaru> <JQ.0.pl.dir.n.nnm> <huDugaru> <N.0.pl.dir.0> <iddaru > <VM.0.pl.3.pr.sim.dcl.fin.n> <.> <PU>

ii) <aa> <DAB.sg.obl..dst.0> < huDugaru> < N.0.pl.dir.0> <bahaLa> <JJ> < jaaNaru> < N.0.pl.dir.0 > <.> <PU>

iii) <avaralli> <PRP.mas.pl.3.dir.0.n.dst.loc> <obbanige> <JQ.0.sg.dir.n.dat> < haNNu> <N.0.sg.dir.0.n> <bahaLa> <JJ> < ishTa> <?> <.> <PU>

iv) <avanu> <PRP.mas.sg.dir.0.n.dst.o> <oLLeyavanu> <?> <.> <PU>

v) <ellaru> <?> < avaniMda < PRP.mas.sg.dir.0.dst.inst> kelasavannu> < N.0.sg.dir.0.n .acc> < maaDisikoLLuttiddaru> <?> <.> <PU>

vi) <elleariMda> < ?> > <avanu> < PRP.mas.sg.dir.0.dst.inst> <hogaLalpaDuttiddanu> <?>

vii) <ella> <JJ> < kelasakkuu> < ?> < avanee> < ?> < beeku> <?> <.> <PU>

viii) <avanu> < PRP.mas.sg.dir.0 > < kelasa> < N.0.sg.dir.0.n > <maaDuva> <JJ> < riiti> < N.0.sg.dir.0.n > <eraDaneeya> < JQ.0.sg.dir.n > <huDuganiguu> < ?> > < ishTa> <?<PU>

ix) <maDabeekaada> <?> < kelasagaLa> < N.0.pl.dir.0.gen> > < paTTi> < N.0.sg.dir.0.n > < maaDiiDuttiddanu> <?> <.> <PU>

Box 3.24. IL-POST Tags for Input Text in Box 3.21

i) <oMdu > < QC> < uurinalli> <NN> <ibbaru> < QC> > < huDugaru> <NN> <iddaru> <VM> -<.> <SYM>

ii) <aa> <DEM> < huDugaru> < NN_PSP> <bahaLa> <INTF> < jaaNaru> < NN> <.> <SYM>

iii) <avaralli> <PRP> <obbanige> <QC> < haNNu> <NN> <bahaLa> <INTF> < ishTa> <?> -SYM>

iv) <avanu> <PRP> <oLLeyavanu> <JJ>

v) <ellaru> <QF> < avaniMda< PRP> kelasavannu> <NN> > < maaDisikoLLuttiddaru> < ? > <.> <SYM>

vi) <elleariMda> <QF> < hogaLalpaDuttiddanu> <?>

vii) <ella> <QF> < kelasakkuu> <NN_PSP> < avanee> < PRP_erg> < beeku> <VM> <.> <SYM>

viii) <avanu> < PRP> < kelasa> <NN> <maaDuva> <VNF> < riiti> < NN> <eraDaneeya> < QO> <huDuganiguu> < NN > < ishTa> <?> <.> <SYM>

ix) <maDabeekaada> <VNF> < kelasagaLa> < NN > < paTTi> < NN> < maaDiiDuttiddanu> <?>

Box 3.25. IIIT-Hyderabad Tags for Input Text in Box 3.21

i) <oMdu >< N-CARD-NHUM-NOM>< uurinalli><N-COM-COU-N.SL-NOM> < ibbaru> <N-CARD-HUM-NOM> < huDugaru><N-COM-COU-M.PL-NOM><iddaru><V-IN-FI><.><PUN-SEP-FUS>

ii) <aa><ADJ-DEM< huDugaru>< N-COM-COU-M.PL-NOM> <bahaLa><ADJ-INTF>< jaaNaru>< N-COM-COU-M.PL-NOM ><.><PUN-SEP-FUS>

iii) <avaralli><PRO-PER-DIST-P3-M.PL-LOC> <obbanige><N-CARD-HUM-DAT>< haNNu><N-COM-COU-N.SL-NOM> <bahaLa><ADJ-INTF>< ishTa><V-DEF><.><PUN-SEP-FUS>

iv) <avanu> <PRP-PER-DIST-P3-M.SL-NOM><oLLeyavanu><ADJ-ABS-PRO.avanu>RP

v) <ellaru><N-CARD-HUM-MF.PL-NOM>< avaniMda< PRP-PER-DIST-P3-M.SL-ABL>kelasavannu><N-COM-COU-N.SL-ACC>< maaDisikoLLuttiddaru><V-TR-CAU-REF-DUR-PST-M.PL><.><PUN-SEP-FUS>

vi) <elleariMda>< N-CARD-HUM-MF.PL-ABL> < hogaLalpaDuttiddanu><V-TR-INF-PASS-DUR-M.SL>

vii) <ella><ADJ-QNTF>< kelasakkuu><N-COM-COU-N.SL-DAT-INCL_clitic>< avanee>< PRP-PER-DIST-P3-M.SL-NOM-EMP_clitic< beeku><V-DEF><.><PUN-SEP-FUS>

viii) <avanu>< PRP-PER-DIST-P3-M.SL-NOM >< kelasa><N-COM-COU-N.SL-NOM> <maaDuva><V-TR-PST-RP>< riiti>< N-COM-COU-N.SL-NOM> <eraDaneeya>< ADJ-ORD><huDuganiguu>< N-COM-COU-M.SL-NOM >< ishTa><V-DEF><.><PUN-SEP-FUS>

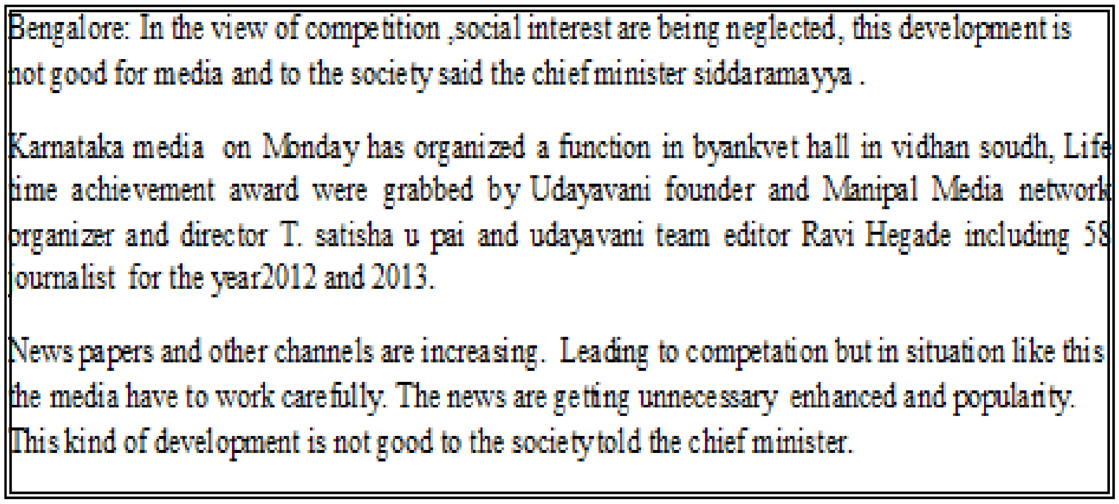
ix) <maDabeekaada><V-TR-INF-MOD-PST-RP>< kelasagaLa>< N-COM-COU-N.PL-NOM >< paTTi>< N-COM-COU-N.SL-NOM> < maaDiiDuttiddanu><V-TR-AUX-DUR-P3-M.SL>

Box 3.26. Our HPOS Tags for Input Text in Box 3.21.

Sample collected from Kannada website is shown in below box, and the English version, transliterated version are shown in following below boxes.

ಬೆಂಗಳೂರು: ಪರಸ್ಪರ ಪೈಪೋಟಿಯಲ್ಲಿ ಸಾಮಾಜಿಕ ಹಿತಾಸಕ್ತಿಗಿಂತ ವೈಯಕ್ತಿಕ ಹಿತಾಸಕ್ತಿಗಳು ಹೆಚ್ಚು ಪ್ರಾಮುಖ್ಯತೆ ಪಡೆದುಕೊಳ್ಳುತ್ತಿರುವುದುಮಾಧ್ಯಮ ಕ್ಷೇತ್ರ ಮತ್ತು ಸಮಾಜದ ದೃಷ್ಟಿಯಿಂದ ಅಹಿತಕರ ಬೆಳವಣಿಗೆ' ಎಂದು ಮುಖ್ಯಮಂತ್ರಿ ಸಿದ್ದರಾಮಯ್ಯ ಕಳವಳ ವ್ಯಕ್ತಪಡಿಸಿದರು. ಕರ್ನಾಟಕ ಮಾಧ್ಯಮ ಅಕಾಡೆಮಿ ಸೋಮವಾರ ವಿಧಾನಸೌಧ ಬ್ಯಾಂಕ್ವೆಟ್‌ಹಾಲ್ ನಲ್ಲಿ ಹಮ್ಮಿಕೊಂಡಿದ್ದ ಸಮಾರಂಭದಲ್ಲಿ ಜೀವಮಾನದ ಸಾಧನೆಗಾಗಿ 'ಉದಯವಾಣಿ'ಯ ಸಂಸ್ಥಾಪಕ ಹಾಗೂ ಮಣಿಪಾಲ್ ಮೀಡಿಯಾ ನೆಟ್‌ವರ್ಕ್‌ನವ್ಯವಸ್ಥಾಪಕ ನಿರ್ದೇಶಕರಾದ ಟಿ. ಸತೀಶ್ **ಯು**. ಪೈಹಾಗೂ 'ಉದಯವಾಣಿ' ಸಮೂಹ ಸಂಪಾದಕರಾದರ ವಿ ಹೆಗಡೆ (ವಾರ್ಷಿಕಪ್ರಶಸ್ತಿ) ಸೇರಿದಂತೆಒಟ್ಟು ೫೮ ಪತ್ರಕರ್ತರಿಗೆ ೨೦೧೨ ಮತ್ತು ೨೦೧೩ ನೇವಾರ್ಷಿಕಪ್ರಶಸ್ತಿ ಪ್ರದಾನ ಮಾಡಿ ಅವರು ಮಾತನಾಡಿದರು. ಪತ್ರಿಕಾ ಆವೃತ್ತಿಗಳು ಮತ್ತು ಸುದ್ದಿವಾಹಿನಿಗಳ ಸಂಖ್ಯೆಹೆಚ್ಚುತ್ತಿದ್ದು, ಇದರಿಂದಸ್ಪರ್ಧೆಏರ್ಪಟ್ಟಿದೆ. ಆದರೆ, ಇಂತಹ ಸಂದರ್ಭದಲ್ಲೇ ಮಾಧ್ಯಮಗಳು ಹೆಚ್ಚುಎಚ್ಚರದಿಂದ ಕೆಲಸಮಾಡಬೇಕಾದ ಅವಶ್ಯಕತೆ ಇದೆ. ಸಾರ್ವಜನಿಕ ಹಿತಾಸಕ್ತಿಗಳೇ ಮುಖ್ಯವಾಗಬೇಕು. ಆದರೆ, ಪೈಪೋಟಿಯಲ್ಲಿ ವೈಯಕ್ತಿಕ ಹಿತಾಸಕ್ತಿಗಳೇ ಹೆಚ್ಚುತ್ತಿದ್ದು, ಸುದ್ದಿಗಳು ವೈಭವೀಕರಣಗೊಂಡು ನಿರಂತರ ಪ್ರಚಾರ ಪಡೆದುಕೊಳ್ಳುತ್ತಿವೆ. ಇದು ಸಮಾಜಕ್ಕೆ ಹಾಗೂ ಮಾಧ್ಯಮಕ್ಷೇತ್ರಕ್ಕೆಒಳ್ಳೆಯದಲ್ಲ ಎಂದು ಹೇಳಿದರು.

Box 3.27. Sample of Text Collected From Website KannadaYahoo.com


Bengalore: In the view of competition ,social interest are being neglected, this development is not good for media and to the society said the chief minister siddaramayya .

Karnataka media on Monday has organized a function in byankvet hall in vidhan soudh, Life time achievement award were grabbed by Udayavani founder and Manipal Media network organizer and director T. satisha u pai and udayavani team editor Ravi Hegade including 58 journalist for the year2012 and 2013.

News papers and other channels are increasing. Leading to competation but in situation like this the media have to work carefully. The news are getting unnecessary enhanced and popularity. This kind of development is not good to the societytold the chief minister.

Box 3.28 English Translation of Text given in Box 3.27.

beMgaluuru paraspara saamajika hitasakigiMta vaiyuktika hitasaktigaLuheccu pramukhyatatepaDedukoLLuttiruvadu maadhyama ksheetra mattusamaajada drshTiyiMda ahitakar beLavaNige eMdu muKhyamaMtri siddaramayya kaLAvaLA vyaktapaDisidaru. karnaTaka maadhyama akaDemi soomavaara vidaanasoudhada byaMkveTa haalanalli hammikoMDidda samaraMbadalli jivamaanada saadanegaagi udayavaaNiya saMsThapaka haagu MaNi PaaL miiDiya neTavarkana vyavasthaapaka nirdes`akaraadaTi. Satis`a. yu.pai haagu udayavaaNi samuuha saMpaadakaraada ravi hegaDe seridaMte oTTu 58 patrakartarige 2012 mattu 2013 ne vaarshika pras`asti pradhaana maaDi maatanaaDidaru. patrikaa aavRuttigaLu mattu suddi vaahinigaLa saMkhye heccuttiddu, idariMda sparde erpaTTide. aadare iMtaha saMdarabadalle maadhyamagaLu eccaradiMda kelasa maaDabeekaada avas`yakate ide. Saarvajanika hitasaktigaLee mukhyavaagabeeku. Aadare paipoTiyalli vaiyuktika hitasaktigaLee heccuttiddu, suddigaLu vaibhavikaraNagoMDu niraMtara prachaara paDedukoLLuttive. Idu samajakke haagu maaDhyama kshetrakke oLLeyadalla eMdu heLidaru.

Box 3.29 Transliteration of Text given in Box 3.27

```
beMgaLuuru<NP.0.sg.obl.gen>:<PU><paraspara><JJ.0.0.dir><paipoTiyalli>NC.0.sg.obl.loc><saa
majika><JJ.0.0.dir><,hitaasaktigiMta>?>vaiyuktika<JJ.0.0.dir>hitasaktigaLu<NC.0.pl.obl.gen>hec
cu <JJ.0.0.dir>praamukhyate< NC.0.sg.obl.>paDedukoLLuttiruvadu <?>maaDhyama <
N.0.sg.dir.0.n>ksheetra<N.0.sg.dir.0.n>mattu<CSB>samaajada<N.0.sg.dir.0.gen.>dRushTiyiMda<
N.0.sg.dir.0.ndat>ahitakara<JJ.0.0.dir>beLavaNige<N.0.sg.dir.0.n.dat>eMdu<?>mukhyamaMtri<N
.0.sg.dir.0.n.>siddaraamayya<NP.mas.sg.obl.gen>kaLavaLa<N.0.sg.dir.0.n>vyaktapaDisidaru<VM
.0.sg.3.pst.dcl.fin><.><PU> karnaTaka<NP.0.sg.obl.gen>maaDhyama<N.0.sg.dir.0.n>akaaDEmi
<N.0.sg.dir.0.n>soomavaara<NC.0.sg.dir.0.n>vidhanashaudada<N.0.sg.dir.0.n.gen>byaMkveTa<?
>haalnalli<?>hammikoMdidda<?>samaaraMbadalli<N.0.sg.dir.0.n.loc>jiivamaanada<
N.0.sg.dir.0.n .gen>saadhanegaagi <N.0.sg.dir.0.n.
pur>udayavaaNi<NP.0.sg.obl.gen>saMsthaapaka<N.mas.sg.dir.o.n.>haagu
<CSB>maNipaal<NP.0.sg.obl.gen>miiDiyaa<?>neTavarkana<?>vyavasthaapaka<nirdhes`akaraad
a<?>Ti.Satis`ayu.pai <NP.mas.sg.obl>haagu <CSB>udayavaaNi<NP.0.sg.obl.gen >samuuha<
N.0.sg.dir.0.n.>saMpadakaraada<?>ravihegaDe<NP.mas.sg.obl.gen>seeridaMte<?>oTTu<JJ.0.sg.d
ir> 58<?>patrakartarige < N.0.pl.dir.0.n dat> 2012<?> mattu<CSB> 2013nee<?>vaarshika
<JJ.0.0.dir>pras`asti<N.0.sg.dir.0.n>pradhaana<N.0.sg.dir.0.n.>maaDi<VM.0.0.0.0.nfn.0>avaru<P
PR.mas.sg.3.dir.0.n>maatanaaDidaru<VM.0.sg.3.pst.dcl.fin><.><PU>patrikaa
<JJ.0.0.dir>aavRttigaLu< N.0.pl.dir.0.n >mattu<CSB> suddi< N.0.sg.dir.0.n > vaaHinigaLa<
N.0.pl.dir.0.n .gen>saMkhe< N.0.sg.dir.0.n
>heccuttiddu<VM.0.0.0.0.nfn.0><,><PU>idariMda<PPR.0.sg.obl.INST>spardhe<N.0.sg.dir.0.n>er
apaTTide<VM.0.sg.3.pst.dcl.fin><.><PU>. aadare<CSB><,><PU>
iMtaha<J.0.0.dir>saMdarbhadalle< ?>maadhyamagalu< N.0.pl.dir.0.n >heccu<JJ.0.0.dir>
ecchradiMda< N.0.sg.dir.0.n .inst>kelasa< N.0.sg.dir.0.n >maaDabeekaada<?>avas`akate
<N.0.sg.dir.0.n>ide <VAUX.0.sg.3.prs.dcl.fin.n><.><PU>sarvajanika <JJ.0.0.dir> hitaasaktigaLee
<?> mukhyavaagabeeku <?><.><PU>aadare<CSB><,><PU>paipoTiyalli <NC.0.sg.obl.loc>
yuktika <JJ.0.0.dir>  hitasaktigaLee<?>heccuttiddu <VM.0.0.0.0.nfn.0> suddigaLu
<N.0.pl.dir.0.n>vabhavikaranegoMDu <?>niraNtara<RB><JJ.0.0.dir>
prachaara<N.0.sg.dir.0.n>paDedukoLLuttive<?><.><PU>idu<PPR.0.sg.3.dir.0.n.prx>samajakke<
N.0.sg.dir.0.n.dat>haagu<CSB>maaDhama<N.0.sg.dir.0.n>kshetrakke<N.0.sg.dir.0.n..dat>oLleyad
alla<?>eMdu<VM.0.0.0.0.nfn.0>heLidaru<VM.0.sg.3.pst.dcl.fin><.><PU>
```

Box 3.30 Website Text KannadaYahoo.com Tagged using IL-POST Tags

beMgaLuru<N-PRP-LOC-NOM><:><PUN-SEP-COL>paraspara<ADJ-ABS> paipoTiyalli><N-COM-COU-N.SL-LOC> saamaajika<ADJ-ABS> hitasaktigiMta <N-COM-COU-DAT> vaiyuktika <ADJ-ABS> hitasaktigaLu>< N-COM-COU-N.PL-NOM>heccu<ADJ-QNTF>praamukhete < N-COM-UNC-NOM>paDedukoLLUttiruvadu <V-TR-CJP-REF-DUR-GRND>madhyama <N-COM-COU-N.SL-NOM>kshetra <N-COM-COU-N.SL-NOM> mattu<CONJ-COOR> samaajada <N-COM-COU-N.SL-GEN> dRshtiyiMda< N-COM-COU-ABL>ahitakara <ADJ-ABS >beLavaNige< N-COM-COU-DAT> eMdu <CONJ-COOR>mukhyamaMtri< N-COM-COU-NOM> siddaraamayya<N-PRP-PER-M.SL-NOM>kaLavaLa <N-COM-COU-N.SL-NOM> vyaktapaDisidaru<V-TR-PASS-CAU-M.PL><.><PUN-SEP-FUS>karnaaTaka<N-PRP-LOC-NOM> maadhama <N-COM-COU-N.SL-NOM>akaaDemi <N-COM-COU-N.SL-NOM>soomavaara <N-LOC-TIM-ABS-NOM>vidhasoudada< N-COM-COU-GEN> byaMkveTa<? >haalanalli< ?> hammikoMdidda <V-TR-PST-RP>samaaraMbhadalli <N-COM-COU-N.SL-LOC> jiivamaanada <N-COM-COU-N.SL-GEN>sadhanegaagi <N-COM-COU-N.SL-PUR> 'udayavaaNiya<N-PRP-N.SL-NOM> saMsthaapaka N-COM-COU—M.SL-NOM> haagu<CONJ-SUB>maNipaala<?> miiDiyaa<? >neTavarkana<? >vyavasthaapaka <N-COM-COU-M.SL-NOM >nirdes`akaraada< N-COM-COU-M.SL-HON-NOM> Ti. Satis`a .yu. pai<N-PRP-PER-M.SL-NOM>haagu <CONJ-SUB>udayavaaNi<N-PRP-N.SL-NOM>samuuha<N-COM-COU-N.SL-NOM>saMpaadakaru<N-COM-COU-M.SL-HON-NOM>V–IN-PST-RP> ravihegaDe<N-PRP-PER-NOM>seeridaMte<V-IN—PST-RP-SIM>oTTu<ADJ-QNTF>58<N-PRP-NUM >patrakartarige <N-COM-COU-M.PL-DAT> 2012<N-PRP-NUM>mattu <CONJ-SUB> 2013nee<?>varshika<ADJ-ABS> pras`asti <N-COM-COU—N.SL-NOM>pradhaana <N-COM-UNC-N.SL-NOM>maadi<V-TR-PST-CJP>avaru<PRO-PER-P3-M.PL-NOM>maatanaaDidaru<V-TR-PST-M.PL><.> <PUN-SEP-FUS>patrikaa <ADJ-ABS> avRttigaLu<N-COM-COU-N.PL-NOM>mattu <CONJ-SUB>suddi <N-COM-COU-N.SL-NOM>vaahinigaLa <N-COM-COU-GEN>saMkhe< N-COM-COU-NOM>heccuttiddu<V-TR-PST-CJP><,><PUN-SEP-COMA> idariMda<PRO-PER-P3-N.SL-ABL> sparade< N-COM-COU-NOM> erapaTTide <V-IN-PST><.>< <PUN-SEP-FUS>aadare<CONJ-COOR>, <PUN-SEP-FUS>iMtaha<ADJ-ABS> saMdarbadalle <N-COM-COU-N.SL-LOC-EMP> maadhyamagaLu <N-COM-COU-N.PL-NOM>heccu<ADJ-QNTF>ωeccharadiMda<N-COM-UNC-N.SL-NOM>kelasa <N-COM-COU-N.SL-NOM>maaDebeekaada<V-TR-INF-MOD-RP> avas`akate <N-COM-COU-N.SL-LOC> ide<V-IN-PST><.><PUN—SEP-FUS> saarvajanika<ADJ-ABJ>hitaasaktigaLee<N-COM-COU-P.SL-NOM-EMP> mukhavaagabeeku<V-DEF-AUX-MOD><.><PUN-SEP-FUS> aadare<CONJ-COOR><,><PUN-SEP-COMA> paipoTiyalli <N-COM-COU-N.SL-LOC> vaiyuktika <ADJ-ABS>hitasaktigaLee <N-COM-COU-N.PL-NOM-EMP >heccuttiddu<V-IN-PRES-CJP-AUX-PST-CJP><,><PUN-SEP-FUS> suddigaLu <N-COM-COU-N.PL-NOM>vaibhavikaraNagoMDu<N>V-REF-PST-CJP>niraMtara<ADV-MAN>prachara<N-COM-COU-N.SL-OM>paDedukoLLuttive<V-TR-REF-PRES-N.PL><,><PUN-SEP-FUS> idu<PRO-P3-N.SL> samaajakke <N-COM-UNC-DAT>haagu <CONJ-COOR>maaDhama< N-COM-COU-N.SL-NOM> kshetrakke<N-COM-COU-N.SL-DAT>oLLeyadalla<ADJ-ABS>N.PRO.adu-NEG> eMdu <CONJ-COORD>heeLidaru<V-TR-FI-PST-M.PL><.><PUN-SEP-FUS>

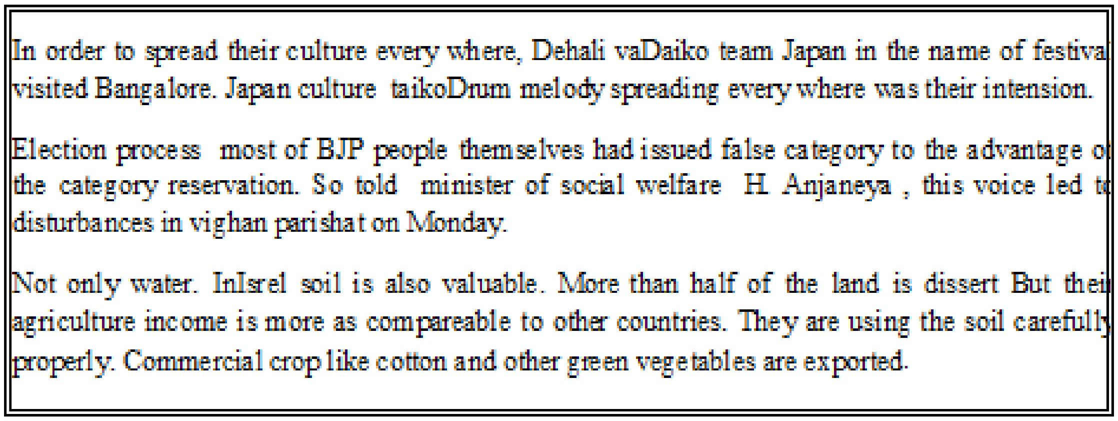
Box 3.31. Website Text from KannadaYahoo.com Tagged using HPOS Tags

```
<beMgaLuuru><NNP>:<SYM>paraspara<JJ>paipoTiyalli<NN_PSP>saamaajika <JJ>
hitasaktigiMta<NN_PSP>vaivyuktika<JJ> hitasaktigaLu<NN_PSP>heccu <QF>
pramukhyate<NN>paDEdukoLLuttiruvadu<VGNN>maadhyama<NN>kshetra<NN-
>mattu<CC>samaajada<NN_PSP>dRshTiyiMda<NN_PSP> ahitakara<JJ>beLavaNige
<NN>eMdu<CC>mukhyamaMtri<NN>siddaraamayya<NNP>kaLavaLa<NN>vyaktapaDisi
daru<VM>.<SYM>karnaaTaka<NNP>maadhyama<NN>akaaDemi<NN>soomavaara
<NST> vidhanasoudada  <NN_PSP>byaMkveTa<?>halanalli<?>hammikoMdidda
<VGNF> samaaraMbadalli <NN_PSP>jivamaanada<NN_PSP>saadhanegaagi<NN-
_PSP>udayavaaNi <NNP>saMstahapaka<NN>haagu<CC>maNipaala <NNP>miDiyaa
<?>neTvarkan <?> vyavasthpaka <NN>nirdeshararaada<?>ti.satiisha.yu.pai<NNP> haagu
<CC> udayavaaN I <NNP>  samuuha <NN>saMpadakaraada><?>ravihegaDe<NNP>
seeridaMte<?> oTTu<QF>58<?>p atrakartarige<NN_PSP>2012 <?>mattu<CC>
2013nee?> vaarshika<JJ>prashasti<NN>pradhaana <NN>maaDi <VGNF>avaru
<PRP>maatanaaDidaru <VM>. <SYM>patrikaa<JJ>aavRttigaLu<NN_PSP>
mattu<CC>suddi <NN>vaahinigaLa<NN_PSP>saMkhe<NN>heccuttiddu<VGNF>,
<SYM> idariMda <PRP_PSP>sparade<NN>erapaTTide<VM><SYM>
aadare<CC>,<SYM> iMtaha<JJ>saMdarbadalle<?> maadhyamagaLu <NN_PSP> heccu
<QF> eccaradiMda  <NN_PSP>kelasa<NN>maadabeekada <?>avas`hakathe
<NN>ide<VM>.<SYM>sarvajanika<JJ>hitasaktigaLee<?P>muKhyavaagabeeku<?>.
<SYM>aadare<CC>, <SYM>paipoTiyalli<NN_PSP>vaivyktikaka  <JJ>hitasaTigaLee<?>
heccuttiddu<SYM> suddigaLu<NN_PSP>
vaibhavikariNagoMDu<?>niraMtara<RB>prachaara <NN> paDedukoLLuttive
<VM><SYM> idu<PRP>samaajakke<NN_PSP>haagu<CC>maaDhyama <NN-
>kshetrakke<NN_PSP>oLLeyadalla<?>eMdu<CC>heLidaru<VM>. <SYM>
```

Box 3.32. Website Text from KannadaYahoo.com Tagged using IIIT-H Tags

ತಮ್ಮsaMpradaayavannuಎಲ್ಲೆಡೆಹಬ್ಬಿಸುವuddes`adiMdaದೆಹಲಿವಡ್ಕೈಕೋತಂಡಜಪಾನ್ ಹಬ್ಬದ ನೆಪವಾಗಿ ಬೆಂಗಳೂರಿಗೆ ಬಂದಿತ್ತು. ಜಪಾನಿನಸಾಂಪ್ರದಾಯಿಕಶೈಲಿಯಟೈಕೋಡ್ರಮ್ ನಾದವನ್ನುಎಲ್ಲೆಡೆ ಪಸರಿಸುವ ಬಯಕೆ ಇವರದು. ಚುನಾವಣೆಪ್ರಕ್ರಿಯೆಯಲ್ಲಿಪರಿಶಿಷ್ಟಜಾತಿ ಹಾಗೂಪರಿಶಿಷ್ಟವರ್ಗಗಳ ಮೀಸಲುಸೌಲಭ್ಯಪಡೆಯಲು ಬಿಜೆಪಿ ಸದಸ್ಯರೇ ಸುಳ್ಳುಜಾತಿಪ್ರಮಾಣ ಪತ್ರನೀಡಿದ್ದಾರೆ' ಎನ್ನುವಸಮಾಜಕಲ್ಯಾಣಸಚಿವಎಚ್.ಆಂಜನೇಯಅವರಹೇಳಿಕೆ, ವಿಧಾನಪರಿಷತ್‌ನಲ್ಲಿಸೋಮವಾರಗದ್ದಲಕ್ಕೆಕಾರಣವಾಯಿತು. ನೀರಷ್ಟೇಅಲ್ಲ, ಇಸ್ರೇಲಿನಲ್ಲಿಮಣ್ಣುಕೂಡಅಮೂಲ್ಯ. ಅರ್ಧಕ್ಕಿಂತಹೆಚ್ಚುಪ್ರದೇಶಮರುಭೂಮಿ. ಆದರೆಅಲ್ಲಿನಕೃಷಿಉತ್ಪನ್ನಗಳಇಳುವರಿಯಾವುದೇದೇಶಕ್ಕಿಂತ ಹೆಚ್ಚು. ಅಪರೂಪಕ್ಕೆಸಿಗುವ ಮಣ್ಣನ್ನುತಂದು, ವಿವೇಚನೆಯಿಂದ ಬಳಸಿ ವ್ಯವಸಾಯನಡೆಯುತ್ತಿದೆ. ಹತ್ತಿಯಂಥ ವಾಣಿಜ್ಯಬೆಳೆಯಿಂದ ಹಿಡಿದು ಸೊಪ್ಪಿನವರೆಗೆಎಲ್ಲ ಬಗೆಯಬೆಳೆಬೆಳೆದು ರಫ್ತುಮಾಡಲಾಗುತ್ತದೆ.

Box 3.33. Text Collected from PrjaavaaNi News Paper



In order to spread their culture every where, Dehali vaDaiko team Japan in the name of festiva visited Bangalore. Japan culture taikoDrum melody spreading every where was their intension.

Election process most of BJP people themselves had issued false category to the advantage o the category reservation. So told minister of social welfare H. Anjaneya , this voice led t disturbances in vighan parishat on Monday.

Not only water. InIsrel soil is also valuable. More than half of the land is dissert But thei agriculture income is more as compareable to other countries. They are using the soil carefull properly. Commercial crop like cotton and other green vegetables are exported.

Box 3.34.English Translation Text Collected from PrjaavaaNi News Paper

tamma sampradaayavannu elleDe habbisuva uddes`adiMda dehali vaDaikootaMDa japaana habbada nepavaagi beMgaLuurigebaMdittu..japaanina saMpradaayika sa`liya Taikoo Dram naadavannuelleDe pasarisuvabayake ivaradu.

cunaavaNeprakriyayalliparis`ishtajaati haagu paris`ishTavargagaLamiisalu saulab hyapaDeyalu bijepi sadasyaree suLLu jaati pramaaNa patra niDiddaareennuva samaaja kalyaNa sacivaeca.aaMjaneeyaavaraheLikevidhanaparishatnallisoomavaara gaddalakke kaaraNavaayitu.

niirashTeallaisrelanallimaNNukuuDaamulya.ardhakkimta heccu prades`a marubhumi aadareallinaKRushi utpannagaLa iLuvari yavudeedees`akkiMta heccu. aparuupakke siguva maNNannutaMdu, vivecaneyiMda baLasi vvyavasaaya naDeyuttide. hattiyaMthavaanijyabeLeyimdahiDidu soppinavarege ellabageyabeLe beLedu raptu maaDalaaguttade.

Box 3.35. Transliteration of Text Collected from PrajaavaaNi News Paper

tamma <PPR.mas.sg.3.dir.ref.n> sampradaayavannu < N.0.sg.dir.0.n .acc>elleDe<NST> habbisuva <?>uddes`adiMda << N.0.sg.dir.0.n .dat> dehali<NP.0.sg.obl.gen> vaDaikoo<NP.0.sg.obl.gen>taMDa < N.0.sg.dir.0.n> japaana <NP.0.sg.obl.gen>habbada< N.0.sg.dir.0.n .acc nepavaagi< ?> beMgaLuurige <NP.0.sg.obl.gen>baMdittu <VM-.0.0.0.fin>..japaanina <NP.0.sg.obl.gen> saMpradaayika <JJ.0.0.dir>sa`liya < N.0.sg.dir.0.n .gen> Taikoo Dram <?>naadavannu < N.0.sg.dir.0.n .acc> elleDe <?>pasarisuva< VM-.0.0.0.nfn>.bayake< N.0.sg.dir.0.n > ivaradu<?><.><PU>

cunaavaNe< N.0.sg.dir.0.n.>prakriyayalli< N.0.sg.dir.0.n .loc>paris`ishta<JJ.0.0.dir>jaati< N.0.sg.dir.0.n> haagu<CSB> paris`ishTa <JJ.0.0.dir>vargagaLa< N.0.sg.dir.0.n> miisalu <JJ.0.0.dir>saulabhya< N.0.sg.dir.0.n>paDeyalu<VM-.0.0.0.inf>bijepi<NP.0.sg.obl.gen> sadasyaree< ? > suLLu< N.0.sg.dir.0.n > jaati< N.0.sg.dir.0.n > pramaaNa< N.0.sg.dir.0.n> patra< N.0.sg.dir.0.n> niDiddaare<VM-.0.sg.3.pst..fn><.><PU> ennuva<<VM-.0.0.0.nfn>.> samaaja<N.0.sg.dir.0.n>kalyaNa<N.0.sg.dir.0.nsaciva<jj.0.0.dir>eca.aaMjaneeya <NP.0.sg.obl. gen> avara <PPR.mas.sg.3.dir.0.n.dst>heLike< N.0.sg.dir.0.n .loc><,><PU>vidhanaparishatnalli< N.0.sg.dir.0.n .loc>soomavaara <NST>gaddalakke< N.0.sg.dir.0.n .dat> kaaraNavaayitu<.><PU> niirashTe<?>alla<NEG><,><PU>isrelanalli<NP.0.sg.obl.gen>maNNu<N.0.sg.dir.0.n >kuuDa<PP><amulya.<JJ.0.0.dir>ardhakkimta<JQ.0.0.dir>heccu<JQ.0.0.dir>prades`a< N.0.sg.dir.0.>marubhumi<N.0.sg.dir.0.n>aadare<CSB>allina<NST>KRushi<N.0.sg.dir.0.n>ut pannagaLa<N.0.pl.dir.0.n>iLuvari<N.0.sg.dir.0.n>yavudee<?>dees`akkiMta<?>heccu.<JQ.0.0. dir>aparuupakke<N.0.sg.dir.0.n.dat>siguva<<VM-.0.0.0.nfn>.> maNNannu< N.0.sg.dir.0.n .acc>taMdu<VM-.0.0.0.nfn>., vivecaneyiMda < N.0.sg.dir.0.n .inst> baLasi<<VM-.0.0.0.nfn>.>vvyavasaaya< N.0.sg.dir.0.n>naDeyuttide<VM-.0.0.0.fin> hattiyaMtha <?>vaanijya <JJ.0.0.dir> beLeyimda< N.0.sg.dir.0.n.inst>hiDidu<VM-.0.0.0.nfn>soppinavarege<?>ella<JQ.0.0.dir>bageya <JJ.0.0.dir>beLe <N.0.sg.dir.0.n> beLedu <VM-.0.0.0.nfn> raptu<N.0.sg.dir.0.n>maaDalaaguttade<?><.><PU>

Box 3.36. PrajaavaaNi News Paper tagged Using IL-POST tag et

tamma<PRP>sampradaayavannu<NN_PSP>elleDe<NST>habbisuva<VGNF>uddes`adiMda <NN-_PSP>dehali<NNP> vaDaiko<NNP>taMDa<NN_PSP> japaan <NNP> habbada<NN_PSP> nepavaagi<NN_PSP>beMgaLuurige<NNP> baMdittu <VM>. japaanina <NNP>saMpradaayika <JJ> s`ailiya <NN>TaikooDram <?> naadavannu <NN_PSP>elleDe<NST> pasarisuva<VGNF> bayake <NN> ivaradu <?><.><SYM> cunaavaNe<NN>prakriyeyalli<NN_PSP> paris`ishTa<JJ> jaati <NN-haagu<CC> paris`ishTa<JJ> vargagaLa<NN>miisalu<NN> saulabhya<NN> paDeyalu <VGINF> bijepi<NNP> sadasyaree   <? >suLLu<NN> jaati<NN>pramaaNa<NN> patra <NN->niDiddaare<VM>ennuva <VGNF> samaaja<<NN>kalyaNa<JJ>sachiva <NN> eh.aMjaneeya<NNP> avara <PRP>keelike <NN><,><SYM> vidhana <NN> parishatnalli <NN_PSP>soomavaara<NN> gaddallakke<NN_PSP> kaaraNavaayitu
<?><.><sym>niirashTee<?>,alla<neg><,><sym>isrelanalli<NNP> maNNu<NN> kuuDa <CC> amulya<JJ>. ardhakkiMta<QF>heccu <QF>prades`a <NN>marabhumi<NN_PSP>.aadare<CC> allina<NST> kRshi<NN> utpannagaLa<NN>iLuvari<NN> yaavudee  <?>des`akkiMta<NN_PSP> heccu<QF><.><SYM>aparuupakke<NN_PSP> siguva<VGNF>maNNannu <NN-_PSP>taMdu<VGNF><,><SYM> vivechaneyiMda <NN_PSP> baLasi <VGNF>vyavasaaya<NN> naDeyuttide<VM><.><SYM> hattiyiMda <? >vaaNijya<NN> beLeyiMda <NN_PSP>hiDidu<VGNF> soppinavarege <NN_PSP> ella <QF>bageya<JJ> beLe<NN>beLedu<VGNF> raptu <NN> maaDalaaguttade<?><.><sym>

Box 3.37. PrajaavaaNi News Paper Text tagged by IIIT-H Tag

tamma<PRO-REF-P23.MFN.PL-GEN ->sampradaayavannu<N-COM-COU-N.SL-ACC->elleDe<ADV-PLA>habbisuva<V-IN-CAU-RP>uddes`adiMda<N-COM-COU-N.SL-ABL> dehali<N-PRP-LOC>vaDaiko<N_N-PRP-ORG>taMDa<N-COM-COU-N.SL-NOM>japaan<N-PRP-LOC>habbada<N-COM-COU-N.SL-GEN>nepavaagi<ADV-DE_N>beMgaLuurige<N-PRP-LOC>baMdittu<V-IN-PST-N.SL>. japaanina<N-PRP-LOC>saMpradaayika<ADJ-DESC->s`ailiya<N-COM-COU–N.SL -NOM>TaikooDram <?>naadavannu<N-COM-COU-N.SL-NOM>elleDe<ADV-PLA>pasarisuva<V-TR-CAU-FUT-RP>bayake<N-COM-COU-N.SL-NOM->ivaradu<PRP-PER-P3-M.SL>PRO.adu.?><.><PUN-SEP-FUS> cunaavaNe<N-COM-COU-N.SL-NOM> prakriyeyalli <N-COM-COU-N.SL-LOC >paris`ishTa<ADJ ABS>jaati<N-COM-COU-N.SL-NOM>haagu<CONJ-COOR>paris`ishTa<ADJ-ABS>vargagaLa<N-COM-COU-N.PL-GEN>miisalu<ADJ-ABS>saulabhya<N-COM-COU-N.SL-NOM>paDeyalu<V-TR-INF>bijepi<N-PRP-ORG>sadasyaree<N-COM-COU-M.PL-NOM-EMP>suLLu<N-COM-COU-N.SL-NOM>jaati<N-COM-COU-N.SL-NOM>pramaaNa <N-COM-COU-N.SL-NOM->patra<N-COM-COU-N.SL-NOM>niDiddaare<V-TR-PST-M.PL> ennuva<V-IN-FUT-RP>samaaja<<N-COM-COU-N.SL-NOM>kalyaNa<N-COM-UNC-N.SL-NOM>sachiva<N-COM-COU-M.SL-NOM>eh.aMjaneeya <N-PRP-PER-M.SL-NOM>avara<PRO-PER-P3-M.SL-NOM>keelike<N-COM-COU-N.SL-NOM><,><PUN-SEP-COMA> vidhana<N-COM-COU-N.SL-NOM>parishatnalli<N-COM-UNC-N.SL-NOM >soomavaara <N-LOC- TIM-ABS-NOM>addallakke<N-COM-UNC-N.SL-NOM>kaaraNavaayitu<N-COM-COU-N.SL-NOM=>V+aagu+ PST- N.SL><.><PUN-SEP-FUS>niirashTee<N-COM-UNC-CLIT-EMP>,alla<ADV-NEG><,><PUN-SEP-COMA> isrelanalli <N-PRP-LOC>maNNu<N-COM-UNC-N.SL-NOM>kuuDa<CONJ-COOR>amulya<ADJ-ABS>ardhakkiMta<N-CARD-COMPF>heccu<ADJ-QNTFF>prades`a<N-COM-COU-N.SL-NOM>marabhumi<N-COM-COU-N.SL-NOM>aadare<CONJ-COOR>allina<N-LOC-PLA-GEN>kRshi<N-COM-UNC-N.SL-NOM->utpannagaLa<N-COM-COU-N.PL-NOM>iLuvari<N-COM-UNC-N.SL-NOM>yaavudee<PRO-INTG-P3-N.SL-NOM>des`akkiMta<N-COM-COU-N.SL-NOM>heccu<ADJ-QNTF><.><PUN-SEP-FUS>aparuupakke<N-COM-UNC-N.SL-DAT->siguva<V-IN-FUT-RP>maNNannu<N-COM-UNC-N.SL-ACC>taMdu<V-TR-CJP><,><PUN-SEP-COMA>vivechaneyiMda<N-COM-UNC-N.SL-NOM>baLasi<V-TR-CAU-PST-CJP>vyavasaaya<N-COM-UNC-N.SL-NOM-->naDeyuttide<V-IN-PRES-N.SL><.><PUN-SEP-FUS>hattiyaMtha<N-COM-UNC-N>SL-SIM>vaaNijya<ADJ-ABS>beLeyiMda<N-COM-COU-N>SL-ABL>hiDidu<V-TR-CAU-PST-CJP>soppinavarege<N-COM-COU-N.SL-PP>ella <ADV-PLA>bageya <ADJ-ABS>beLe<N-COM-COU-N.SL-NOM>beLedu<V-TR-CAU-PST-CJP>raptu<N-COM-UNC-N.SL-NOM->maaDalaaguttade<V-TR-INF-MOD-PASS-PRES-N.SL><.><PUN-SEP-FUS>

Box 3.38 PrajaavaaNi News Paper Text tagged by HPOS Tags

From the synthetic sample text, 16 words remained untagged from the yahoo text and 12 words remained untagged from the PrajaavaaNi text. It is observed that IL-POST tags are capturing more morpho-syntactic information than the IIIT-H tag set. The disadvantage of IIIT-H tag set is, it captures shallow information. Using HPOS tagset all words from the synthetic sample and prajaavaaNi text are tagged but 5 words remained untagged in the yahoo text, this was because the yahoo had English words, hence those remained un-tagged. The tagging efficiency is shown in below figure.

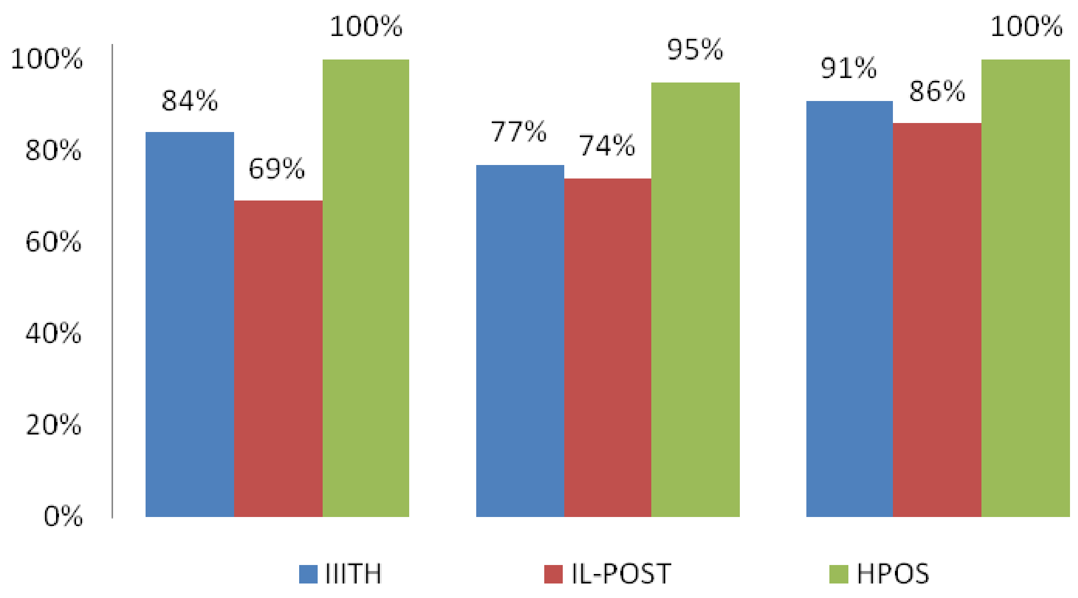


Figure 3.4. Tagging efficiency By different Tagsets

Table 3.11. TaggingRate Using Different TagSets

Tag Set	Synthetic (39 Words)	Yahoo Text (109 Words)	PrajaavaaNi (89 words)
IIITH	33 tagged	83 tagged	81 tagged
IL-POST	27 tagged	91 tagged	77 tagged
HPOS	39 tagged	104 tagged	89 tagged

Tagging rate is more in HPOS tagset as compared to other two tagsets. However it is assumed that the lower the granularity, the greater the accuracy. But it has been shown by researchers on the CRATER project that a tagset of higher granularity may actually give better results. The precision and recall of different tagsets on different samples is shown in table 3.12.

Table 3.12. Showing Precision versus Recall

Performance Measures	IL-POST	IIIT-H	HPOS
Precision	82%	88%	99%
Recall	97%	96%	98%
F-measure	88%	91%	98%

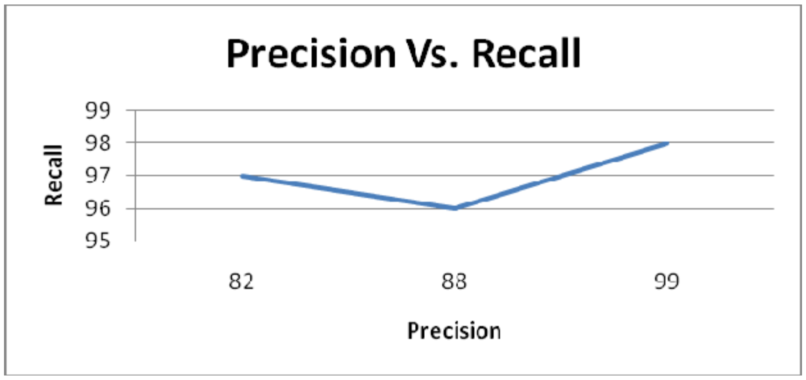


Figure 3.5. Precision versus Recall for 3 Tagsets

From the figure 3.4 it is observed that the tagging rate is more for HPOS than IIIT-H and IL-POST tags. MT applications require fine detailed information about each word. If you simply tag all nouns as NN, information like whether the noun is masculine/ feminine/neuter/ singular /plural all these items of information are not obvious from IIIT-H tag set.

But IL-POST are useful in MT application since they are fine grained and capture more information but fail to handle clitic information, new compound verbs, auxiliary and modal auxiliaries and contingent forms. Hence efficiency may be affected.

HPOS tags are useful in MT applications and are fine grained and handle clitic, modal auxiliaries, conjunct verbs etc., and tagging efficiency is more here as compared to other two tag sets.

We performed separate tagging process for verbs by choosing the corpus from CIIL.Few selected sample files from CILL corpus. It was observed that the CILL corpus has more complex verb formations as compared to the yahoo text. Verbs are formed by adding causative, reflexive, aspect auxiliaries and modal auxiliaries forms and also clitic forms. That may be the reason for low tagging rate in case of IL-POST tagset. Though these aspects are not handled in IL-POST, however IIIT-H tagging rate is more as compared to IL-POST but the tag set is flat in nature. The verb tagging on CIIL samples by different tagsets is shown in figure 3.6.

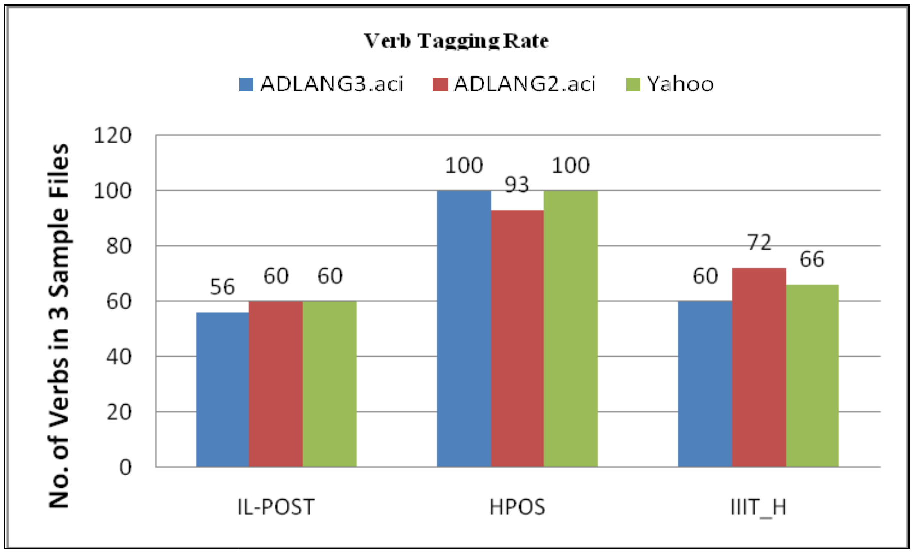


Figure 3.6. Verb Tagging Rate by Different Tagsets

From the figure 3.6 it is observed that the tagging rate for HPOS tags is comparatively higher than the other two tag sets. The number of verbs tagged by different tag groups is shown in table 3.13.

Table 3.13. Tagging Rate Using Different TagSets

Tag Set	ADLANG2.aci Sample for top 50 sentences (33 verbs)	ADALANG3.aci (104 verb forms) Sample for top 50 sentences	YAHOO (15 verbs) Sample for top 50 sentences
IIIT-H	24 tagged	63 tagged	10 tagged
IL-POST	20 tagged	59 tagged	9 tagged
K-HPOS	33 tagged	94 tagged	15 tagged

Upon manual inspection of ADLANG3.aci file we observe that the types of verbs that occur in the corpus are formed from derivative stems and hence detailed classification is required to capture the minute detail, without looking into the fact of more number of tag sets. Here also HPOS performs better than the other tags sets.

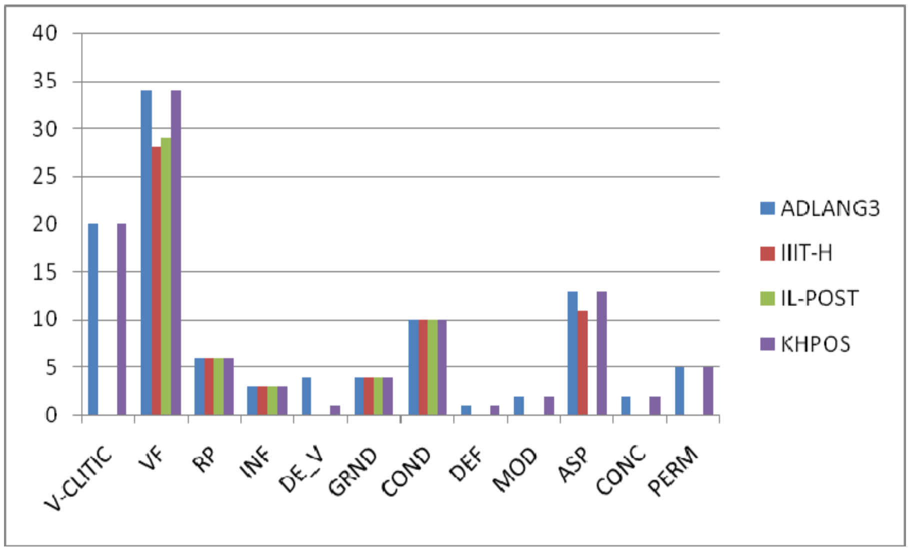


Figure 3.7. Verb Tagging rate on ADLANG3.aci File

ADLANG 3.aci file had a complex verb stem, it had a combination of causative,reflexive and clitic stems derivations. From the figure 3.8 it is observed that HPOS tagging rate is good in capturing more features.

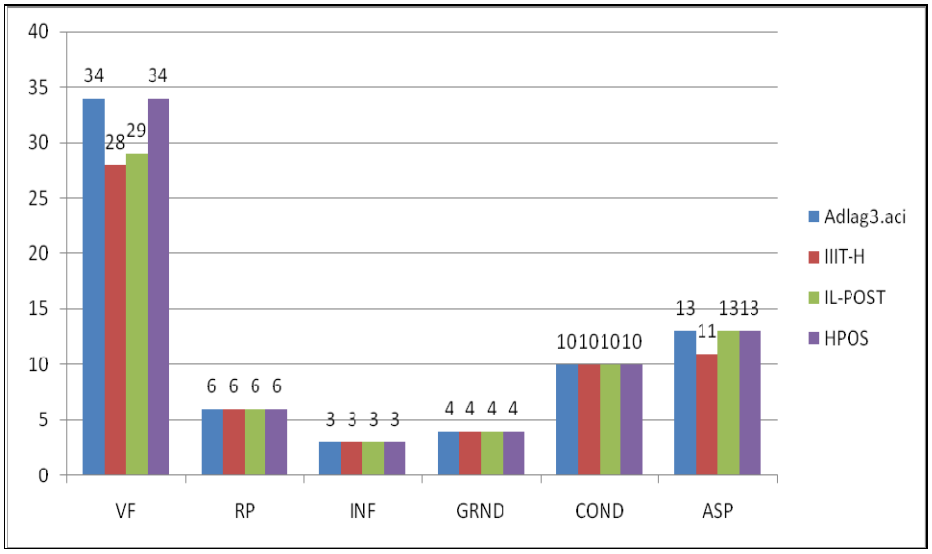


Figure 3.8. Matched Verb Types Across the Tagsets

Upon consideration of whole ADLANg3.aci file. Figure3.8 shows number of forms matched across the tag sets. Around 54 verb forms match across the tag groups out of 104 different verb forms that occur in ADLANg3 file. The identification of different verbs is more in the HPOS tag set, if complex verb forms occur more than naturally the recognition rate will be more for the HPOS tag set. But however there are verb finite forms which occur due to simple verb stems instead of derived or complex root in such cases the three tag sets match more at identification of verb forms.

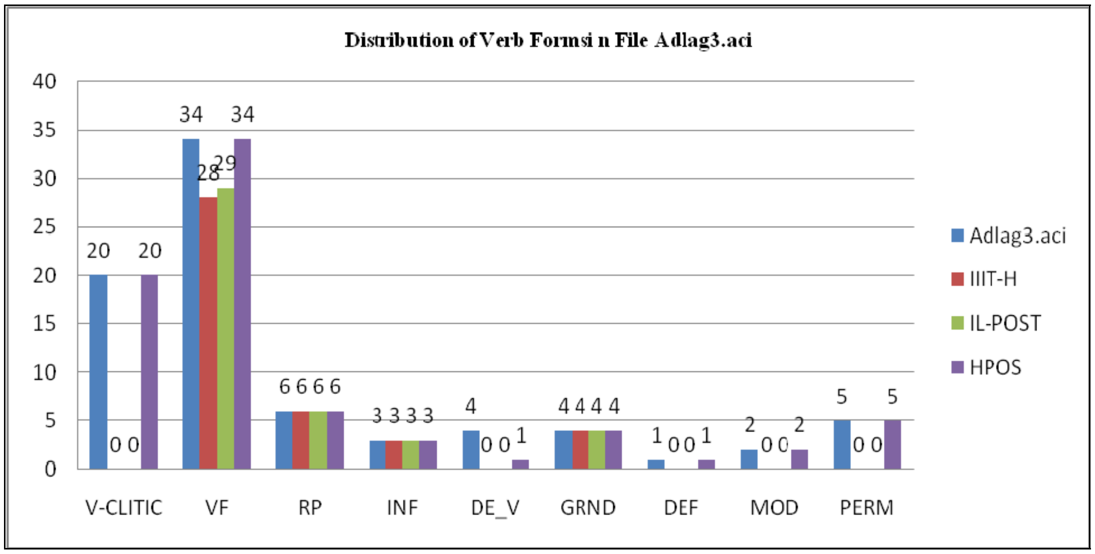


Figure 3.9. Verb Tagging rate on ADLANG2.aci

In ADLANG2.aci, different verb forms occurring are less in number: many forms like hortative, conjunctive, contingent, causative did not occurred, hence tagging rate in all the tagsets are more or less nearer. As shown in figure 3.9.The category-wise distribution of tags in the HPOS tag set is shown in table 3.14.

Table 3.14. Total Atomic Tags In HPOS

Category	No of Atomic Tags
Noun	45
Pronoun	36
Adverb	10
Conjunction	3
Interjection	1
Punctuation	18
Postposition	10

Verbs	42
Adjective	5
Total	**170+1**

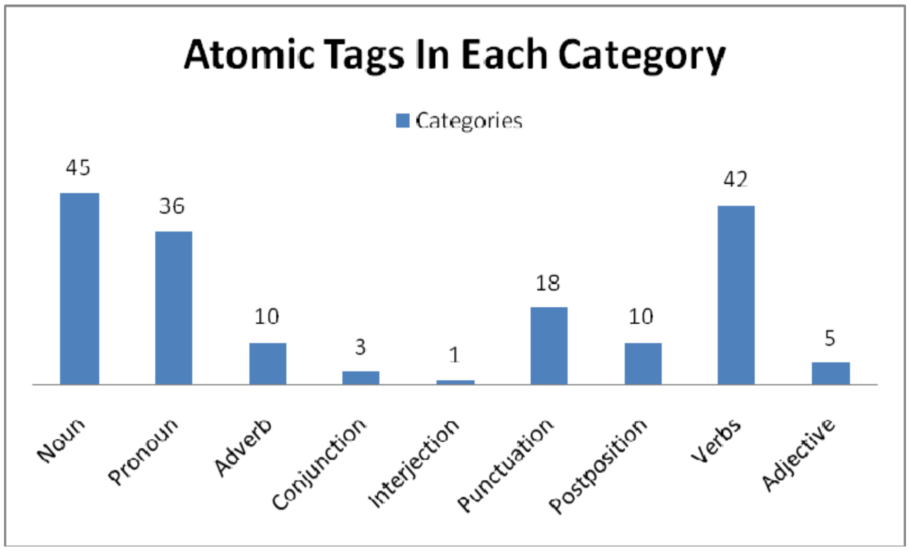


Figure 3.10. Category Wise Distribution of HPOS Tags

Figure 3.10 shows the total number of tags developed category wise. Tags for nouns are more,followed by verbs, pronouns and so-on.

3.7. Difficulty Faced in Designing Tag set

Gerunds – GRND

We were in confusion as to treat gerund as a noun or verb. For example gerunds take case inflection like nouns say maaDuvudannu (doing with accusative marker), and it can be the subject of a sentence just like a noun. But gerunds have tense, aspect inflections like verbs and they are capable of taking their own arguments: for example,tinnuvudu"eating" needs an object. If we treat gerunds as nominal, then nouns having verb features look strange. Later we decided to capture this feature through morph derivation. Morphological analyzer gives the details of derivation from verb to noun. We have solved this open issue existing in Baskaran et.al. IL-POST tagset. For a task such as machine translation, fine-grained part-of-speech tags such as verb, 1st person, and singular, present, indicative etc are necessary. For many other tasks, such as IR (Information retrieval) WSD (word sense disambiguation) coarse POS tags such as v-verb or n-noun are sufficient.

- **Summary**

The objective of the HPOS tagset is to capture in depth detailed information of word grammar which is necessary for many NLP applications. HPOS tagset offers advantages such as flexibility, cross-linguistic compatibility, reusability, and decomposability. The tagset is necessary for annotation of corpora. Identifying morpho-syntactic features, deciding its sub classification, is not an easy job for a language like Kannada due to complex word formation process. *Hence designing morphosyntactic tag set for Kannada verbs is quite challenging.*There was need for an exhaustive tagset. Atotal of 42 tags are designed for Verbs and 128 tags for non verbal categories are developed. Tagging efficiency in HPOS tags is more as compared to IIIT-H and IL-POST tags.Evaluating our hierarchical tagset in further stages of our research work of morphological analyzer and in building the morph related dictionary is the next task in hand.

Chapter 4

Building Morph-Related Electronic Dictionary Using Semi-Automatic Methods

In the previous chapter the importance of developing a hierarchical tagset for Kannada is discussed. In this chapter discusses on how an electronicdictionary can be built within a few months of time. It is mainly focusing on developing a morphology related electronic dictionary. There is a need for an electronic dictionary which gives detailed information about each word, with all features like common noun, countable noun, noun case, gender, number etc. HPOS tagset is used for tagging each word in the dictionary. The existing dictionaries are not directly usable for the present morphological processor work. **Hashing technique** is used for dictionary lookup.

The morphological analyzer and generator (MAG) is developed as part of proposed research work. There was a need of suitable electronic dictionary for the proposed morphological system. The morphology related dictionary keeps the information useful for the morphological generator like marking of **"Real U" information, countable/uncountable feature on nouns, marking of transitivity and intransitivity information,** exceptions and inflections of words whose morphology is not regular and has irregular past participle forms of verbs etc. These items of syntactic information play a crucial role in developing the MAG system. Till today this kind of monolingual dictionary using ***Hierarchical tagset*** is not available for Kannada. This is the first attempt in this direction among Indian languages. A Dictionary of around 30,000 words is developed using semi-automatic method, Pattern matching, filtering technique and intersection operations have been applied. A total of 8898 closed class words are available in this dictionary. *The Department of Electronics (DoE) Central institute of Indian language (CIIL) corpus* is used for developing dictionary. Each word is assigned as an expanded tag giving full description, clear attention and manual effort are must.

The Electronic dictionary is an asset for computational linguistics. The electronic dictionary developing work described here uses semiautomatic methods for building dictionaries from electronic text corpora. A hierarchical tag for tagging each word in the dictionary is used instead of using just a flat tagset. Developing an e-dictionary using hierarchical tag is a valuable effort because each word has to be marked for its gender, number, countable and uncountable features. Till today there is no dictionary exists for Kannada which is developed using the hierarchical tagset. The hierarchical tagset gives detailed morph-syntactic information and is useful in developing Morphological generator and analyzer systems. It is shown here that the available resources are inadequate and full of errors. Simple techniques such as heuristic pattern matching, regular expressions, frequency vs. rank kind of knowledge, stemming and filtering techniques have been used in building this dictionary. A dictionary of 30,000 plus words is constructed. A Total of 126 different cases, word ending patterns are used in developing this dictionary. This work will look at the process of *creating one of the necessary NLP resources in developing a morphological processor.*

4.1 Introduction

A Dictionary is a representation of words and meaning. *The association of word patterns and meanings is not rule governed, hence we have to store words in the dictionary. An electronic dictionary can be better than a printed dictionary. The information can be added more easily in electronic form than the printed dictionary.* Electronic dictionaries provide *greater flexibility and convenience for human users*. The user can alter the contents or change the structure. One can do a lot of things that are necessary for example retrieving of all bisyllabic words, all words of foreign origin, all those words which are both nouns and verbs etc. by writing programs.

Dictionaries can be directly used by a variety of *NLP applications such as spell checking, syntactic analysis or parsing, machine translation (MT), morphological analysis* etc. It is observed that the available dictionaries are inadequate both in terms of coverage and in terms of the capturing detailed information.

An electronic dictionary can grow and respond to changes as the language evolves. Most machine readable dictionaries are manually built by human experts or transformed into electronic forms from hard-copy versions through an expensive digitization process. Dictionaries and thesauri are store houses of knowledge words.

To know the state of art we carried out a literature survey in the following section.

4.2 Literature Survey

Lots of work has been carried out on the development of electronic dictionaries for English as compared to Indian languages. Following is the gist of work cited in the literature.

(Goldberg D. E, 1989) has proposed a genetic algorithm to fuse different linguistic hypotheses and also Machine Learning statistical methods to extract bilingual dictionaries. He used large amounts of parallel texts to build this dictionary. This kind of dictionaries are useful in current corpus based machine Translation application.

(Tokunaga et al.) have proposed automatic thesaurus construction based on grammatical relations among the words in the sentences. The proposed method constructs the thesaurus by using a hierarchical clustering algorithm. In the experiment four RBT (relation based thesaurus) of Japanese nouns were constructed from 26,023 verb-noun co-occurrences.

(Yamamoto) has proposed a method of generating a translation dictionary from Japanese /English parallel texts. In this method, English and Japanese compound noun phases are extracted from parallel texts and searched by matching their possible translations generated by the existing translation dictionary.

(Kumano, A. and Hirakawa) have proposed a method of building an MT dictionary from Parallel Texts Based on Linguistics and Statistical Information. This is useful in machine translation application.

(Patrick Hanks) discuses the relationship between a lexical data base and monolingual dictionaries, the role of corpus evidence and need for vocabulary coverage are discussed in his article "Compiling a Monolingual Dictionary for Native Speakers". This article gives a survey of the main issues confronting compilers of monolingual dictionaries in the age of the internet.

(Kay M andRoscheisen M) have proposed a technique of Text-Translation Alignment using a bilingual dictionary. They have presented an algorithm for aligning texts with their translations that is based only on internal evidence. The relaxation process rests on a notion of which word in one text corresponds to which word in the other text that is essentially based on similarity of their distributions. It exploits a partial alignment of the word level to induce a maximum likelihood alignment of the sentence level, which is in turn used, in the next iteration, to refine the word level estimate. The algorithm appears to converge to the correct sentence alignment in only a few iterations.

(Utsuro, Matsumoto and Nagao) have proposed a unified framework for bilingual texts matching by combining existing hand-written bilingual dictionaries and statistical techniques. The process of bilingual text matching consists of two major steps, i.e. sentence alignment and structural matching of bilingual sentences.
Regarding the Indian Scenario we have a dictionary in Printed form like Mysore Kannada- English dictionary; this dictionary is available in PDF form. It is not possible to process by writing programs. It is meant for end user to look up meaning of an English word in Kannada and is not suitable for present work. For each English word, the corresponding meaning or description is given in Kannada. As such, we do not always find equivalent Kannada words directly.

Another electronic dictionary is available at IIIT –Hyderabad website (IIIT-H, 2008). It is a bilingual dictionary which has both Kannada and Hindi words. Kannada-Hindi dictionary is in ISCII form. It is advantageous as it is in electronically processable form but it is useful only to some extent. Also there are a lot of inconsistencies. Treatment of words differs from our principles. We cannot directly rely on this dictionary for the proposed work and more over the dictionary is tagged with a flat tagset.

There are two more dictionaries available viz. Prof. Krishna dictionary (pc 2009) has 1 lakh words, it is in electronic form but many words belong to old Kannada literature, which are rarely used today, more over the words are tagged with flat tagset. Another dictionary developed by Prof. K .N Murthy, (pc, 2009). This dictionary has some inconsistencies, here noun phrases are treated as noun, this is not correct, because a phrase consists of many words, for example the phrase "aMdaaju vaarshika paTTi" is marked as a noun. Marking collection of words with single tag is not correct.

There is a need of an electronic dictionary which gives detailed information of the word, with all features. We have several organizations in the country seriously engaged in research and development in computational linguistics and NLP. But our lexical knowledge bases in computerized forms are minimal. The importance and urgency of electronic lexicons in India cannot be over emphasized. We are proposing the semi automatic methodology for developing an electronic dictionary for Kannada Language. As there is not much work reported for Kannada towards NLP oriented dictionary development, that is also another motivation to move in this direction. There is not even a single machine in readable form of a dictionary which is in process-able form and which we can make use for our morphological processor work.

4.3 Issues in Building Electronic Dictionary

The three most important questions that have to be answered while developing the dictionary are. **i.**) what kinds of words are listed in the lexicon/dictionary?. **ii)** What information should the entries provide?. **iii)** How should that information be organized and structured?

4.3.1 Contents of an Entry

What information a dictionary should give, is to some extent theory bound. Say, should the lexicon give transitivity, case frames or a subcategorization frame? Should the lexicon say that 'soldier' is '+ Human' '? What, all semantic features should the lexicon include? How should the lexicon denote the selection restrictions? How exactly should the lexicon represent the fact that 'play' is either a noun or a verb in first person (singularor plural), or second person (singular or plural), or third person (plural but not singular)? How should the lexicon capture the generalization that other verbs also behave the same way? What defaults can a lexicon use to avoid saying the same thing over and over? The answers to some of these questions depend on the links between the lexicon and morphology and other parts of programs that may make use of the lexicon. In other words, the morph dictionary consists of a list of words and extra information indicating their morphophonemic behavior where it is required. What about grammatical information?. There seems to be general agreement about such notions as 'noun' and ‘verb’. Organization of information is an important factor.

4.3.2 Organization of the Information in the Lexicon

So, the information that the lexicon should contain is inseparably tied up with the kinds of programs which use it and the human users who use the lexicon/dictionary and also how exactly they wish to use it, and the kind of phonological, morphological, syntactic, semantic and pragmatic knowledge they might make use of. The lexicon may contain all morphemes and only those words which may not be derived from otherwords and productive processes and, those associated pieces of information which are unpredictable and idiosyncratic with respect tothe rules or knowledge available elsewhere in the system. Pragmatic knowledge is usually considered to be the least relevant in the context of a lexicon, but the distinction between lexical or semantic knowledge and general pragmatic knowledgeis hazy. *The only general principle is, therefore, whenever possible and wherever found useful, employ rules to capture generalizations and thus reduce the burden on the lexicon*. In general, the *information contained in an electronic dictionary must satisfy the usual requirements of any computerized information system the information should be complete, nothing essential be left out, nothing should be underspecified or over specified, there should be no resolvable ambiguities, redundancies or circular and endlessly recursive paths.* The information must be authentic, reliable and dependable and it should be structured and organized so as to achieve maximum efficiency both interms of storage space required and in terms of time and effort required to make use of the information for carrying out various NLP tasks. An electronic dictionary may use techniques of computer science to achieve maximum utility with minimum effort.

4.4 Proposed Method of Developing Morphological Dictionary

One might think of starting a new lexicographic project to create anelectronic dictionary. It is obvious that creating a dictionary is a tedious task in terms of manpower, time and financial support required. It also takes a great deal of lexicographic experience to make a good job of it. It would be impracticable and unproductive for computational linguists to set about constructing a dictionary from scratch. Given this scenario, a very promising alternative is to depend on the dictionaries already available in book form and attempt to extract, reorganize and enter the information contained in them into the computer. This way we are making use of the enormous amount of thought, research and experience that have gone into making these existing dictionaries. Hopefully, the information they contain is reliable and authentic apart from being comprehensive. Here method of developing electronic dictionary is divided into 3 phases. In phase 1, we make use of an existing electronic dictionary, in phase 2, we use frequency estimation method, in phase 3, we use pattern matching, stemming, filtering and knowledge of language data base for developing the electronic dictionary using the text corpora. To start with we focus on the study of available resources.

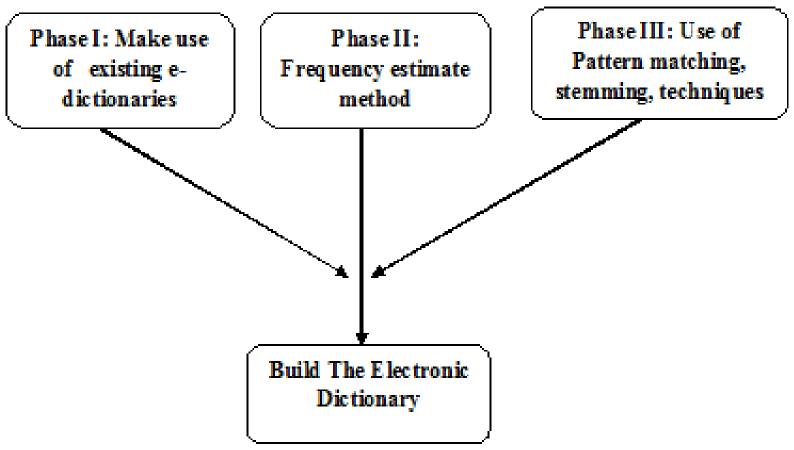

Phase I: Make use of existing e-dictionaries
Phase II: Frequency estimate method
Phase III: Use of Pattern matching, stemming, techniques
Build The Electronic Dictionary

Figure 4.1. Proposed Method of Building an Electronic Dictionary

4.4.1 Phase 1:- Method of Using Available Resources

We have also used available existing electronic Kannada dictionaries to some extent in developing this dictionary. As we cannot use these resources directly due to inconsistencies. We performed intersections operations on POS categories of these dictionaries and manually checked and tagged each word with hierarchical tagset, because the words in these lexicons were tagged with flat tagset. Knowledge of native language helps in designing the heuristics for developing an electronic dictionary.

Available lexical resources including text collections, word lists, dictionaries, etc. must be made best use of while developing new electronic dictionaries. Here we take a quick look at some of the available resources and how exactly we can take advantage of them in our work. In this work we make use of available electronic resources and avoid completely manual methods. Resources in hard copy form. Here we shall see the quality issues and suitability of various electronic dictionaries, text corpora andalso issues taken care while adopting them for use. Their classification of words is sometimes too coarse and does not provide sufficient distinctions between words. There are 3 electronic dictionaries available which can be processed by writing programs these resources are useful to some extent. The available dictionaries are shown in table 4.1, table 4.2 and table 4.3.

Available Electronic Dictionaries included are as follows.

- `IIIT-H, Kannada-Hindi dict: 4 Categories', v, adj, avyaya, p.`

Kannada-Hindi bilingual dictionary is available at IIIT- Hyderabad website in electronic form. But it is in ISCII form. This Kannada-Hindi dictionary has 4 Categories: n, v, adj, avyaya. Kannada-Hindi ISCII dictionary has about 30,000 entries. This is converted into our Roman notation using tool ISCII to roman converter. This dictionary has many inconsistencies. Some suffixes like "gaLeMdare" (plural suffixes gaLu followed by eMdare), gaLaagive etc are treated as avyayas and stored in dictionary, this is not correct. These words are not at all avyayas, iddaane("he is"), iddaaLe ("she is") are treated as avyaya, these are also not avyayas. *Consider an example entry in IIIT-H dictionary.*

a) **sadaacaari,"avyaya","avy",1. sadaacaarii/sus`iila/s`ubha -acaraNavaalaa.**

The above entry is a noun but treated as an avyaya. This is not correct. Another word haudu (yes) is treated as a noun. According to our grammar it is an adverb. Caution is needed in using resources of this kind. Moreover the tag is flat not hierarchical as proposed in our designing methodology.

- Prof. K N Murthy's Dictionary

5 Categories:

Verb- v, Noun -n:Adjective -a,Pronoun -pr:Conjunction –c. Simple closed class words like athava, aadare are missing in this dictionary, but stillwe can make best use of it because it is in electronic form which can be processed bywriting programs. Here also there are many inconsistencies, like consider the following entry ***lex (vaarSika aayavyayada aMdaajupaTTi, [[n]])*** is a noun phrase, the meaning is annual budget estimate the list is treated as a noun, this is not correct but however can be tokenized into vaarSika, aayavyayada, aMdaaju, and Patti can be assigned different tags by decomposing. There is no adverb category, interjection category in this dictionary, no sub categorization in the main categories. Our design principles of tagging dictionary words are not satisfied.

Prof. V Krishna's Dictionary (Personal Communication).
Prof. V Krishna's dict has 9 Grammatical Categories:
Adjective -adj, Adverb -adv: Interjection -intj,
Conjunction -conj: Preposition -prep, Pronoun -pron:
Indeclinable - ind, Noun -n Verb -v:

This dictionary is available in electronic form. This dictionary has 10,0000 plus entries. There are many inconsistencies.This dictionary is quite useful.It is available in the form, word slist followed by category. This dictionary consists of words used in the early days of Kannada literature. For examplewords like aaDalabu, aacaamla, aacamana, aaDeela, etc. are nounsbut these are rarely usedin modern Kannada. Simple adverbs like, mellane,"slowly" are not present. The words have a flat tag, minute distinctions required for hierarchical tagging are not available here.

The description of the 3 available dictionaries is shown in tables 4.1, 4.2 and 4.3. This dictionary is converted to our transliteration form.

Table 4. 1. Prof. KNM Dictionary

Category	No. of words
Nouns	9251
Verbs	839
Adjectives	349
Pronouns	65
Conjunction	1

Table 4. 2. IIIT-H. Kannada-Hindi Dictionary

Category	No. of Words
Nouns	21622
Verbs	4479
Adjective	1545
Avyaya	2901
Pronoun	19

Table 4.3. Prof. V Krishna Dictionary

Category	No of Words
Verbs	183555
Nouns	104345
Interjection	188
Pronoun	101
Preposition	11
Adjectives	9127
Conjunction	30
Indeclinable	71
Adverb	1877

- ***English-Kannada Mysore University Dictionary***

This dictionary is available in PDF form. It is not possible to process by writing programs. It is meant for the end user to look up the meaning of an English word in Kannada and is not suitable for ourwork. For each English word, the corresponding meaning or description is given in Kannada. As such, we do not always find equivalent Kannada words directly.

We have developed an algorithm for processing these dictionaries and tried to extract the words, category wise and tried to find the common words among these by taking intersection using hashing technique to store words and their tag. We got around 7054 words from all these dictionaries to be sure cases, later we checked manually all these words and tagged them with hierarchical tag principles.

4.4.1.1 Algorithm for processing electronic Dictionaries

Input: Read the available Dictionary

Output: add words from the existing Dictionaries to our dictionary.

Start

Step 1:Read dictionary File

Step2: Preprocess text by splitting on space, removing unwanted characters like? -/<> () Etc. and store in input file, if blank line do not process it.

Step 3: While (not end of file) do

Step 4: Check its encoding For ISCII or UNICODE

Step 5: Convert it into our Transliteration form using Iscii toRomanconvertor program (ir.pl)or Unicode to Roman convertor program (ur.pl).

Step 6: For each POS category in the available dictionary First grab yourself a wordlist of nouns in dictionary

Ex:**cat < monolingual dictionary > | grep "-n"'|awk –F{print $2\}.** // command for extracting nouns, similarly, extract all head words form the available dictionary and store them in separate files.

Step 7: Repeat the above steps for all the dictionaries

Step 8: For each category-wise file from different dictionaries. Take the intersection of the POS category files. Common words to assure Surety of nouns or verbs or other Category

Step 9: After getting sure of the words category-wise, change tag each word manually as our HPOS (Hierarchical part of speech tag set.

Stop

The process of taking intersection among the words from the available dictionaries is to check the surety of their categories, i.e. how many words treated as noun , adjective, adverb in one dictionary is also treated as noun, adjective, adverb in other dictionary. The processing of existing electronic dictionaries is shown in below flow diagram.

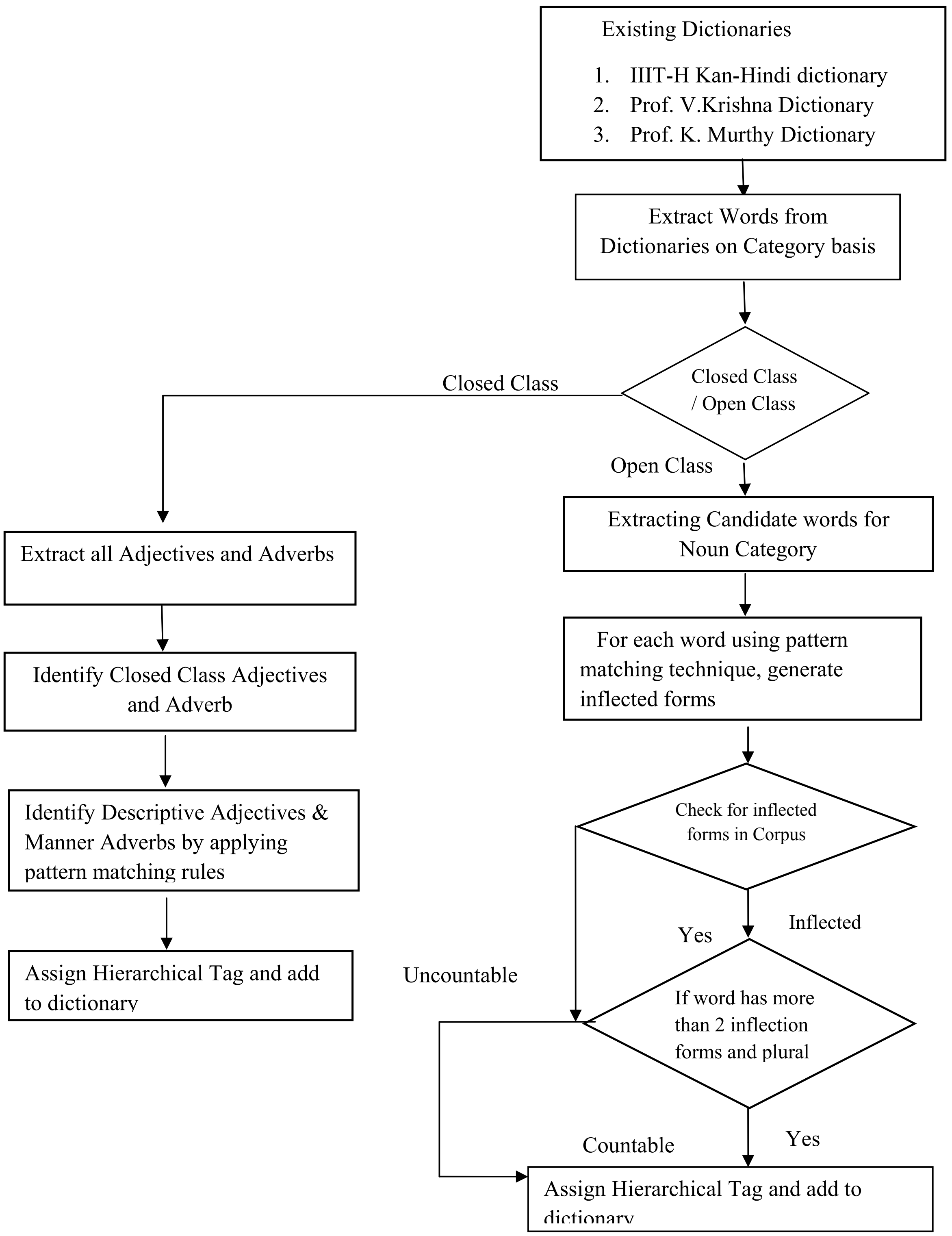

Existing Dictionaries
1. IIIT-H Kan-Hindi dictionary
2. Prof. V.Krishna Dictionary
3. Prof. K. Murthy Dictionary
Extract Words from Dictionaries on Category basis
Closed Class / Open Class
Closed Class
Open Class
Extract all Adjectives and Adverbs
Identify Closed Class Adjectives and Adverb
Identify Descriptive Adjectives & Manner Adverbs by applying pattern matching rules
Assign Hierarchical Tag and add to dictionary
Extracting Candidate words for Noun Category
For each word using pattern matching technique, generate inflected forms
Check for inflected forms in Corpus
Yes
Inflected
Uncountable
If word has more than 2 inflection forms and plural
Countable
Yes
Assign Hierarchical Tag and add to dictionary

Figure 4.2. Processing available Dictionaries

Process of collecting nouns from the existing available dictionaries is not direct addition, but has to be checked for inflections for their surety of POS category. The process is shown in the figure 4.2.

There are formatting inconsistencies in regard to existing dictionaries. Available dictionaries are not directly usable. They require some preprocessing, since they are in different encoding like ISCII or Unicode, with different fonts. It is interesting to see how well these dictionaries agree with regard to their Kannada content. The Table 4.7 shows pair-wise intersections. It is clear that the dictionaries vary widely and none seems to provide wide coverage. The dictionaries differ significantly in terms of the number and nature of grammatical categories and the features they indicate. Thus we see that the available text corpora and electronic dictionaries are far from adequate and we need to develop a wide-coverage electronic dictionary, giving detailed morpho-syntactic tagging. In the next section, we describe our on-going efforts in this direction.

4.4.2 Phase 2:- Method of Using Frequency Estimate Approach

A corpus is a large and representative collection of language material stored in a computer process-able form. The corpus provides the basic language data room from which a variety of lexical resources can be generated. DOE-CILL is in ISCII format, it is converted to roman using translator tools developed by us. Each distinct word form is called a 'type' and each occurrence of a type counts as a ‘token’. Statistical analysis of CIIL corpus (Akshara Bharati et.al, 1998) is shown in Figure 4.3. The figure shows how many new types will be found as the corpus size increases. It can be seen that Kannada, like other Dravidian languages, has a significantly large number of word forms. The type/token ratio is thus 11.29% for Kannada, much higher than for languages such as English (which is about 3.42%) (Akshara Bharati et.al, 1998). Indo-Aryan languages are much closer to English than to Dravidian languages in this respect. Dravidian languages are characterized by a very rich system of morphology and words are long and complex. Type-Token growth rate analysis, Type Token analysis **of DoE-CIIL corpora of size 3 million word is shown in figure 4.3 and in table 4.4.**

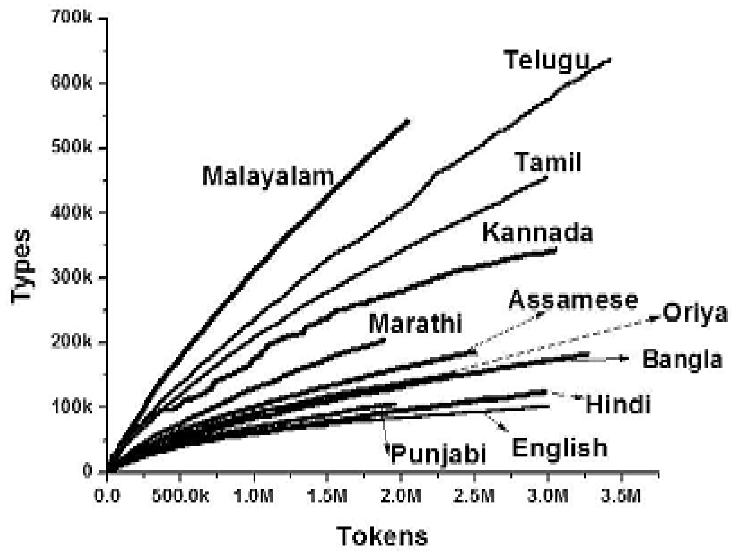


Figure 4.3. Type-Token Rate Analysis of DoE-CIIL Corpora

Table 4.4. Type-Token Rate Analysis of DoE-CIIL Corpora

Language	Family	Type	Token	Type Token Ratio%
English	-	3000000	102690	3.42
Hindi	IA	2980968	124932	4.19
Punjabi	IA	1953762	105602	5.40
Bangla	IA	3270332	183131	5.60
Oriya	IA	2361191	152766	6.47
Assamese	IA	2502687	185616	7.42
Marathi	IA	1886143	204302	10.83
Kannada	DR	3047568	343971	11.29
Tamil	DR	2986273	454070	15.20
Telugu	DR	3669322	636022	17.33
Malayalam	DR	2046207	542239	26.50

We observe that the frequency estimation plays a major role in building the dictionaries. The reason is words should be added in order of frequency is quite intuitive, the higher the frequency, the more likely the word appears in the text that is being translated as stated by Zipf's law. For example in English you can almost be sure that the words like "the" or "a" will appear in all but the most basic sentences.

The higher the frequency of the word, the more you "gain" from adding it. Word frequency vs.Word rank, A plot of word frequency is in log log coordinates. Assuming X is rank of a word in the frequency table and y is the total number of the word's occurrences. Zipf's law corresponds to the upper linear portion of the curve, roughly following the green (1/x) line. We will find interesting property. One is that multiplying the rank of a word by its frequency a number which is pretty constant is obtained. That's called Zipf's Law as shown in figure 4.4. For every 10 words or so you add, it is probably worth going back and repeating this step, especially for highly inflected languages, one lemma produces many word forms and the word list is not lemmatized. Any correspondences produced by this method must be checked by native or fluent speakers of the language in question. It works by first taking all singular forms out of the list, then looking for plural forms, then printing out those which have both singular and plural.

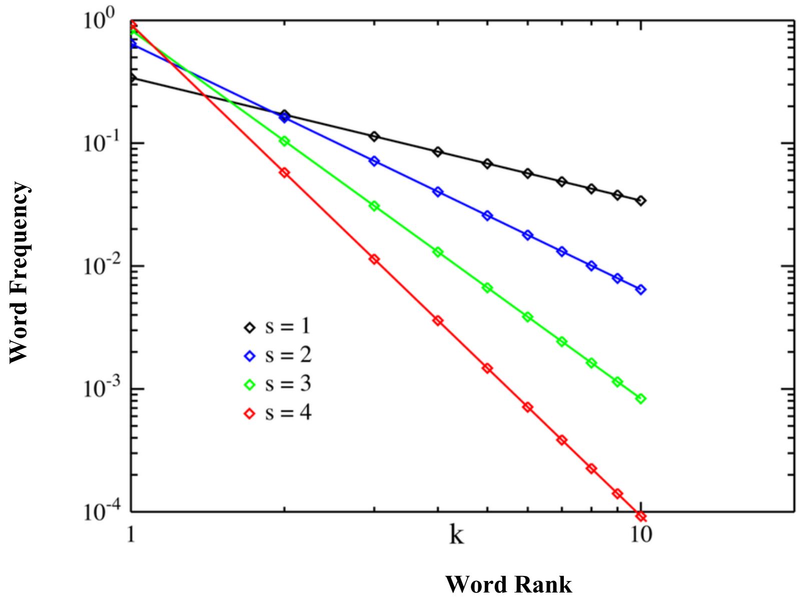


Figure 4.4. Zipf law for Frequency v/s Rank

4.4.2.1 Algorithm for Using Frequency Estimation Method

Input: Raw Corpus File

Output: add words to the dictionary based on frequency estimation method.

Start

Step 1: Read the input file

Step 2: Check its encoding For ISCII or UNICODE

```
Step 3:Convert it into our Transliteration form using
Iscii to Roman convertor program or Unicode to
Romanconvertor program.
Step 4:  Tokenize input File
 Preprocessing of text by splitting on space, removing
unwanted characters like. "? -/<> ()" Etc. and store in
input file, if blank line do not process it.
Step 5: While (not end- of file) do
Step 6: Find the words types and their frequency
  Use the hash technique to store words & their
Frequencies
Step 7: Sort the words list
Step 8: Add words to the dictionary in decreasing order
of frequency since higher frequency words are closed
class words, also look low frequency words and add to our
dictionary, tag each word with HPOS
Stop
```

First all sentences from the corpus are tokenized into words. Frequently used words are in the top, most of these contribute to closed class words. The importance o ffrequency estimates is for the new pairs to have the best possible coverage with a minimum of effort. It is very important to add words and rules in decreasing frequency, starting with the most frequent words and phenomena. The reason that words should be added in the order of frequency is quite intuitive, the higher the frequency, it is more likely to be a functional word. Then word list is sorted, and a table of words and frequency is created. It is quite easy to make a crude collection of words using a simple UNIX command sequence (a single line).

- **cat file.txt | tr '' ''\012| sort -f | unit - c| sort –nr > out.txt**

It is observed that with about 1000 words you may have 75% of the text covered.Once the collection of words is available, it is probably best to skim off the top 20-30,000,in to a separate file.

- **cat out.txt| head- 20000 > top.lista.20000.txt**

We propose that the frequency with which specific words are used in everyday language exerts a general and law-like influence on their rates of evolution, those words which are exceptional i.e. arbitrary in at least one of their various features, will be entered in the lexicon. Use of corpus for building the dictionary plays crucial role.

4.4.3 Phase 3:- Method of Using Pattern matching Techniques

We applied various pattern matching techniques and the stemming techniques to extract the words from the corpus. As a native speaker of Kannada, we had an inherent knowledge of morphotactic rules for Kannada, which helped to a great extent in the development of the dictionary even though developing a dictionary was not an easy job. Nouns in Kannada get inflected for gender, number and cases. *To extract nouns from the corpus many linguistic features like list of possible case suffix, glide insertion at the time of saMdhi, plural suffix information etc. are needed.* A Lot of manual effort is required at the time of stemming when rational and non rational words share a common suffix for number and gender agreements.

A combination of 126 patterns were used to extract nouns from the corpus. Department of electronics (DoE) CILL corpus was used for developing this dictionary. We apply the strongest and surest heuristics first. For example, plural formation in Kannada is nontrivial, we can start with singular forms. Assuming oblique stems (as in genitive case) is the same as the basic stem (in nominative case), we inflect the bases. Upon manual inspection of these cases on a random basis, we concluded that the resulting words were all surely nouns. Thus a dictionary of nouns could be finalized. SaMdhi rules shown in table 4.5 are useful.

Table 4.5. SaMdhi Rules

Glide	Stem Ending			
	A	e/i	O	U
Rational	N	y	N	V (if u is real) (if u is enunciative then u is dropped)
Non Rational	V /d	y	-	(if u is enuciative then u is dropped) v (if u is real)

As a native speaker of Kannada, we have an idea of incremental suffix that occurs when two morphemes conjugate. For example in Kannada the masculine words ending in 'a', has 'n' as glide insertion at the point of conjugation. We have also applied this logic of 'glide insertion' for animate masculine pattern suffix along with increment then do apply stemming to get the basic root form. For neuter gender word 'd' is the increment suffix. Using this differentiation we got many correct words from the corpus, while extracting inflected words from the corpus. Instead of using just the case suffix alone for example 'annu' accusative, we have taken different forms of 'annu' along with its incremental insertions like say vannu, yannu, nannu /Lannu are humans only, vannu/yannu is a shared suffix for human/nonhuman.

Table 4.6. Use of suffix patterns for Extracting Words from Corpus

Masculine Suffix Pattern	Feminine Suffix Pattern
daara: vaarasudaara	gaarti: aMdagaarti
gaara : citragaara	koori : caDikoori
vaMta : niitivaMta	gaati : aMdagagaati
ka : dhanika	tti : bigatti
Iga: gaaNiga	ti: gaLati:
nu : dhiranu	vaMte: buddivaMte
koora: caDikoora	nu:aapaaditaLu

1024 masculine nouns are extracted by pattern matching and 394 feminine words obtained. 3664 adjectives are obtained. Exceptions are stored in the dictionary. For example, the word makkaLu (children) occur in plural form. There is no singular counterpart, some words like halavaru, kelavaru, etc. also occurs as plural forms there is no singular form for these words. Many such words are listed as plural entries in the dictionary since these are exceptions. *Another observation is 600 words are of dual type, representing both feminine and masculine words like "apaaditanu", "apaaditaLu" are obtained during pattern matching. W*e decided to store apaadita in the dictionary under another category as protonoun -PROTN, 'nu' and 'Lu' suffixes are added in the nominative case in noun morphology. Kannada has some irregularities like 'gaLu which is a suffix generally used for nonhuman plural'. The same suffix is used in plural formation of humans also in situations like say janagaLu 'people', in such cases an observation is made. Such words represent all humans without gender specifications. In Kannada we donot have inflection for adjectives and adjectives usually end with letter 'a'. *310 words are both adjective and also (masculine) noun. It is observed that 'ka' ending nouns are both adjective and noun. Say: praamaaNika (honest). 11 case suffixes are used for noun extraction. Accusative suffix is 'nannu', instead of using just annu, we have taken different forms of annu along with its incremental insertions like say vannu, nannu, Lannu, yannu. We know that n is an incremental suffix for masculine, so that we can extract more masculine, feminine words. So a total of 72 case suffix combinations for rational were tried. Noun ending with other patterns like ka,ke,ta, Ta, pu etc., are used for extracting nouns from the corpus, more than 40 such patterns were tried.* It is even possible to define new search criteria when a new resource is added by defining common pointers on searchable information parts. In the case where a normal search returned no results, a reverse lookup was also executed.

4.5 Format of Our Morphological Dictionary

The development of a dictionary goes with a specific requirement. The purpose of our development of a dictionary is, we require a dictionary of the type, lemma followed by its tag. The structure of our morphological dictionary entry is shown in figure 4.5.

Word	Seperator1	Tag1	Tag2	Seperator2	Morph related Information

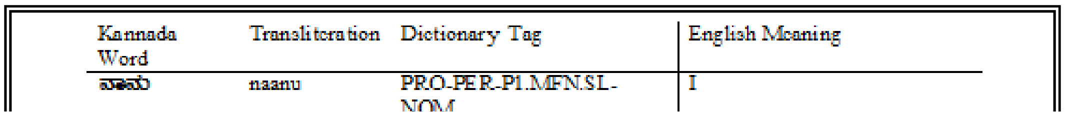
Figure 4.5. Structure of Dictionary Content

Our HPOS Dictionary format:

One entry per line. <Word>||<TAG>||<TAG>: Relevant Morph information

The first field is the root word followed by two vertical bars (||) to separate the word and its tag. If the word has another tag then that tag is also assigned, using another set of vertical bars and if the word carries any grammatical information useful for morphological generator that is also stored in the dictionary by using the separator double colon (::). Grammatical information useful for our morphology system is also stored in the dictionary using the separator: (double colon) to help the morphological generator. In Kannada the words that end with ‘u’ behave differently if word ends with real ‘u’ then glide ‘v’ is inserted during saMdhi. If the word ends with enunciative u then no glide is inserted during saMdhi. We have captured this information in the dictionary. All real u ending words are stored with extra information indicator field. The word guru (teacher) is a word ending with a real ‘u’. But the word niiru (water) even though it ends with u it is not real ‘u’, it is enunciative. It is necessary to store such useful information for the morphological generator. We have stored such morph related information in the dictionary as shown here guru|| N-COM-COU-M.SL-NOM-NULL: REALu.

Another important issue is, if single a word form is representing different meaning. Some word like kaaDige means “to the forest and also ‘kaajal’ (black eye liner). Such words are stored by means of two hierarchical tags like kaaDige|| N-COM-COU-NS-NOM-NULL||N-COM-COU-N.SL-DAT-NULL: ROOT kaaDu. The following box 4.1 shows the sample dictionary.



Kannada Word	Transliteration	Dictionary Tag	English Meaning
ನಾನು	naanu	PRO-PER-P1.MFN.SL-NOM	I

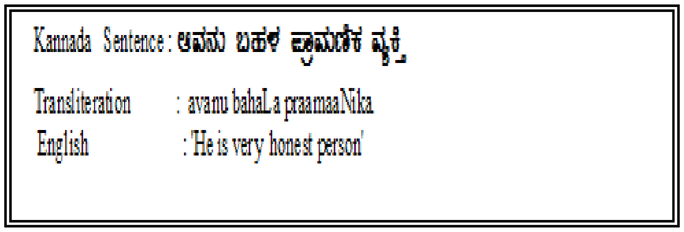
Box 4.1. Sample Morphological dictionary

4.6 Problems Faced in developing the Dictionary

Lots of issues and confusions arise, while assigning proper tag for the words, the following discussions reveal the issues and our decisions regarding the treatment of the words.

i. ***How to tag words like praamaNika in the Dictionary?***

Consider the following sentence in box 4.2.



Kannada Sentence : ಅವನು ಬಹಳ ಪ್ರಾಮಾಣಿಕ ವ್ಯಕ್ತಿ

Transliteration : avanu bahaLa praamaaNika

English : 'He is very honest person'

Box 4.2. Example Sentence for Word praamaNika

If we store **praamaNika** as adjective in the dictionary, we miss the information that it behaves as a noun masculine also. Then when another form of this word, say praamaNikanu occurs in the corpus, how it should be treated was a problem, because the noun follows plural formation like pramaaNikaru.

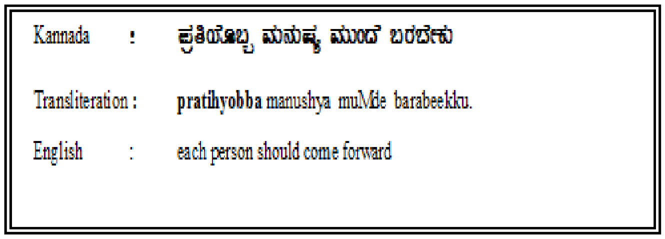
Soln 2: If word praamaaNika "honest" is treated as noun masculine and an adjective, this solution has one problem.When praamaaNikaLu comes as a word, we cannot derive the feminine word from the masculine praamaaNika.

Soln 3 *:* Later we thought of giving adjective, noun masculine, noun feminine all 3 tags to the word praamaaNika, but when we hear the word, it does not seem sensible to store praamaaNika as feminine in the dictionary.

Soln 4: Finally we decided to store it as **protonoun** (a noun in its preform or root with tag PTN) which has adjective behavior. By adding the suffix "nu" we get masculine and by adding Lu, we get feminine, and we have the 'ru' pratyaya adding for plural formation.

ii. ***How to treat words like pratiyobba(each person), innobba(another Person), beerobbaa (different person).***

Soln 1: We included these under pronoun indefinite category and observed the following situations. Consider an example in box 4.3.



Kannada	:	ಪ್ರತಿಯೊಬ್ಬ ಮನುಷ್ಯ ಮುಂದೆ ಬರಬೇಕು
Transliteration	:	**pratihyobba** manushya muMde barabeekku.
English	:	each person should come forward

Box 4.3. Example Sentence for Pratiyobba.

In this example **pratiyobba** is modifying the noun 'manushya' and acts as an adjective. But the basic rules of pronouns say as follows.

Rule 1: A pronoun can be replaced by a noun or pronoun can act in place of a noun.

Rule 2: A pronoun canbe the head of a noun phrase

Rule 3: A pronoun can be inflected by case markers and clitics

Rule 4*:* A pronoun cannot be modified by an adjective. Adjectives cannot come before a pronoun (i e.an adjective can be a noun modifier but not a pronoun modifier).

Rule 5: And above all, pronouns in genitive case are adjectives. Say 'avana mane cennagide', (his house is nice) here avana (his) is a genitive form of pronoun and acts as an adjective. This is an accepted rule.

Rule 6: Pronouns cannot modify a noun.

Drawbacks in this solution: In the above examples, prtiyobba, innobba etc. are in nominative form acting as noun modifiers which is not correct. A pronoun modifying a noun looks strange and violates rule 5 and 6. Rule 1, 2, 3, 4 are satisfied. Hence the above conclusion is not correct.

Soln 2: If we include these under adjective.

- **Basic rules for adjective are.**

Rule 1: Adjective cannot be head of noun phrase.

Rule 2: Adjectives cannot take case inflections and clitics

Rule 3: Adjective can be modified by adjective.

Observations in the solution: if **pratiyobba** is treated as adjective then the word pratiyobba is modifying noun manushya treatment as adjective is okay. But if we consider another form of pratiyobba say “pratiyobbanu baMdanu” (every person came). In this sentence rule 2 fails because pratiyobbanu is inflected form and taking case inflections, now rule 1 also fails because adjective cannot be head of noun phrase. Hence including word pratiyobba under adjective not correct.

Soln 3: If we include word “pratiyobba and its variants under noun.

- **Basic rules for noun.**

Rule 1*:* A noun can replace a pronoun.

Rule 2*:* A noun can be the head of a noun phrase.

Rule 3*:* A noun can be inflected by case markers, clitics.

Rule 4*:* A noun can be modified by an adjective. An Adjective can come before a noun (i.e. adjective acts as a noun modifier).

Rule 5*:* And above all, all nouns in genitive case are adjectives.

Rule 6: A noun can modify a noun.

Observations in the solution: All rules are satisfied according to rule 6. Example in box 4.3 is satisfied, and ‘pratiyobbanu baMdanu.' is also satisfied. Case inflection rule is also satisfied because adjective can not be head of noun phrase. An adjective followed by a pronoun like suMdara innobba (beautiful another person) is not a correct form, whether to include such words under nouns is another question. Later we decided to include them under **noun-cardinal-human**, since noun cardinals also behave the same way. These words are not modified by adjectives.

iii. ***Which form of Noun/Verb is to be stored in the Dictionary?***

Solution: We decided to store the nominative form of a word in the dictionary. For example if we store huDuganu (boy with nominative inflection) in a dictionary as root word and try to derive other forms out of it, it will not work, At the foremost if huDuga 'boy' is a word for analysis, how do we analyze?. We cannot say nominative minus 'nu' plus ‘a’ is the way. This is not according to the grammar of any language. Nominative form is the bare form of any noun hence we decided to store huDuga as dictionary entry. While storing word form in the dictionary it is decided to store a form in the dictionary, from which all other forms of a verb/noun can be generated. For Verbs also the same rule is applied and imperative forms are treated as root forms.

iv. ***Why to Store Inflected Forms in the Dictionary?***

Pronouns are irregular i.e. pronoun morphology is not rule governed: For example 'ru', 'gaLu' are used as plural suffixes in Kannada to generate plural forms. So as accordingly, plural form of the word naanu (I), should be **naanuru or naanugaLu**, similarly 'avana' (his) is genitive form avanu (he) naana (mine) should be the correct genitive form of naanu (I), but this is not the case "nanna" is the genitive form in usage. What is the theory behind consonant germination of 'n' in "**na+n+n+a",** dropping of vowel **'a'** in **naanu**, we still don't have any proof for such derivations, hence we say pronouns are irregular in morphology. It is also observed that there is dative case irregularity, say consider the dative word forms **avanu (he)➔ avanige (to him), naanu (I)➔ naanige (me)** should be the dative form. But we have "nanage" as correct form in dative case instead of naanige. How this derivation occurred is not clear, hence we decided to store inflected forms of all words in the dictionary as exceptions. *Another issue is regarding the locative case and the time noun. All case inflections are not possible, if we leave our morphological system to handle such inflections, morphological generator becomes over general and generates ungrammatical word forms, hence to avoid over generality we have stored inflected forms of such words also in the dictionary.* Consider a locative noun 'alli' (there), we observe that when we add case inflections to this word the word forms maybe like **alliyannu, alliyaoDane, alliyoMdige**, etc. But these are notcorrect forms. Hence we have stored inflected forms of locative nouns in the dictionary, instead of handling them under morphology.

v) Issues with derivational suffix 'vike'

In Kannada we use 'vike' as suffix for deriving nouns form verbs. Say **vibhajisuvike**, 'partitioning'. **vibhajisu** is the verb root, so whether it should bestored in the dictionary or not even though it is derived from the noun 'vibhajane', this is not just addition of 'isu' suffix to the noun, unlike the case of

pres`ne+isu=pras`nisu (questioning). For all verbs suffix “isu” is added as causative suffix and also as a voice modifier while generating morph forms. After adding isu to **vibhajisu (divide)** it becomes ‘vibhajisisu’, which again is an invalid form. All words ending with “isu” are treated separately.

There was another decision conflict whether to store words like **maataaDu** 'speaking' (maatu +aaDu) a noun+verb compound in the dictionary or should be left for morph system to handle. Later it was decided to handle such formations through morph system.Generating of compound verbs is handled in our morph system by adding verbalizes likeaagu iru, aaDu, paDu, goLLu etc.

4.7 Experiments and Results

Lot of experiments to analyze the coverage of developed dictionary with existing dictionaries is performed. Distribution of word types from closed class words is performed. Ambiguity testing for the words in the dictionary is performed. Deciding word category is not an easy task due ambiguities in word forms and also meaning. We have words having more than two tags in the morphological dictionary out of 31,548 words. It is observed that 1418 words are ambiguous i.e. 4.5 %, 39 types of ambiguities are found in the dictionary. Some combinations are frequent, some are rare, 15 combinations of ambiguities have occurred only once. The ambiguity in our dictionary is less compared to other printed dictionaries.

Prabhushankar’s Kannada-English dictionary is in printed form, few pages are selected around 506 words from Prabhushankar dictionary are analyzed for checking ambiguity cases and it is observed that 39 words are ambiguous out of 506 words. The ambiguity is 7%. And we observed that there are some tag inconsistencies. A word like ekaagra is treated as both noun and adjective in that dictionary, according to our tagging criteria ekaagra is an adjective not a noun. *The variation of ambiguity in our case is mainly due to adoption of our tagging principle that, if the word takes the inflection mark as noun, instead as an adjective or adverb or postposition.* That may be the reason for low ambiguity cases in our dictionary as compared to other printed dictionaries.

A Total of 8898 Closed Class Words have been developed by us using different sources. Rules for extraction of words are designed by studying various grammar books like *A Grammatical approach to Kannada Language by Harold Spencer*,A *practical key approach to Kannada by Reverd.F.Ziegler and D.N S Bhat*. We got a total of 7054 words from the existing dictionaries which were checked for accuracy of their categories is shown in table 4.7.

Table 4.7. Words Obtained From Intersection of Dictionaries

Intersection of Dictionaries	**Common Words**
Prof. Krishna Dictionary & IIIT Hyderabad	21334
Prof. Krishna Dictionary & Prof K N Murthy	6235
Prof. K N Murthy & IIIT-Hyderabad	5067
Total Words After Intersection	10054

The words in the language can be generally specified as closed class words and open class words. Open class words increase in number with coining of new words in daily usuage but closed class words are fixed in number and are referred as functional words.

- **Open Class Words**

Open classes of words include content words. The meanings of such words are given in dictionaries. New words often get added and some words may gradually go into oblivion. Here open class words include nouns, verbs, descriptive adjectives and adverbs of manner.

- **Extracting Nouns**

The basic structure of a noun in Kannada is noun+gender+number+case+clitic. Initially, all nouns are extracted from all existing electronic dictionaries. Since we are not very sure about the quality of these dictionaries, we apply a variety of pattern matching and heuristic filters to extract sure cases. Since we are assigning a descriptive tag to each entry in the dictionary lot of manual effort is required at the time of stemming when rational and non rational words share a common suffix for a number of inflections and increments.We generate inflected forms and check for their existence in the DoE-CIIL corpus. If several inflected forms are found, we can take the base form to be a noun for sure.

- **Extracting Verbs**

All verbs are extracted from all available bilingual electronic dictionaries. Since we are not very sure about the quality of these dictionaries, we manually checked all the verbs and 513 verbs were identified as sure. These verbs are not marked for transitive and intransitive nature we manually checked these and tagged each word. We included only verb root in the dictionary since inflected forms can be generated from the root. For example words like gaaDihoDi, which is not an original verb. They form a separate type of compound verbs (noun- verb combination). These words are not included in our verb dictionary. The total number of basic verbs in the dictionary is1001.

- **Coverage Analysis**

Coverage analysis deals with the examination of how much of a corpus/dictionary can be covered by a given set of types. We have done several experiments to check the coverage of proposed dictionary with existing dictionaries is shown in table 4.8 and figure 4.6.

Table 4.8. Coverage of Available Dictionaries with Our Dictionary

Dictionary	Words	Words found from	% coverage

		Our Dictionary	
Prof. K N Murthy	10505	9124	80
IIIT- H (KAN-HIN)	30000	16656	53
Prof. Krishna	104345	15561	0.15

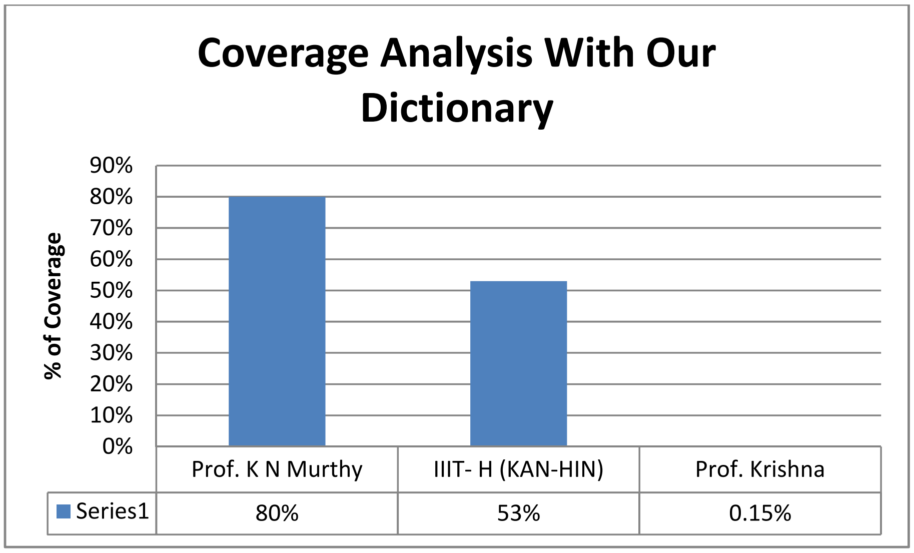


Figure 4.6. Coverage Analysis of other Dictionaries

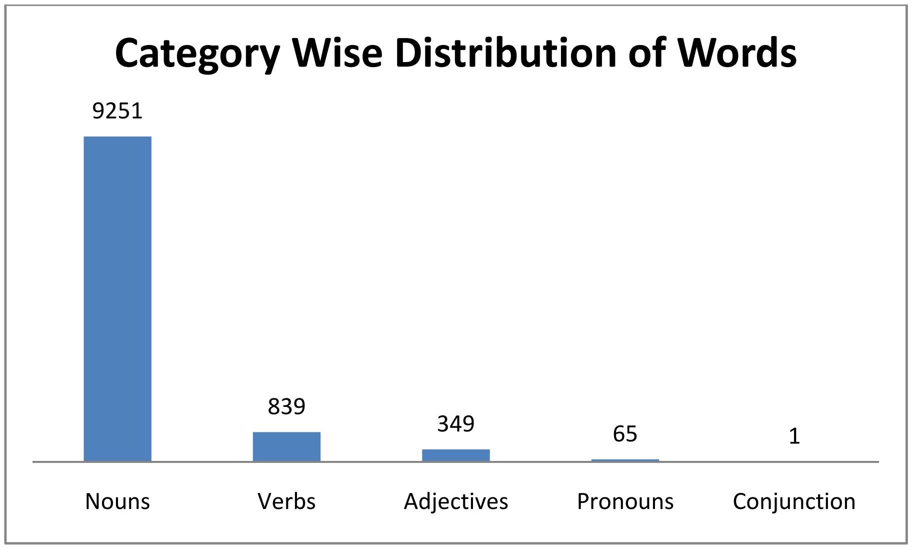


Figure 4.7. Prof. K N Murthy's Dictionary

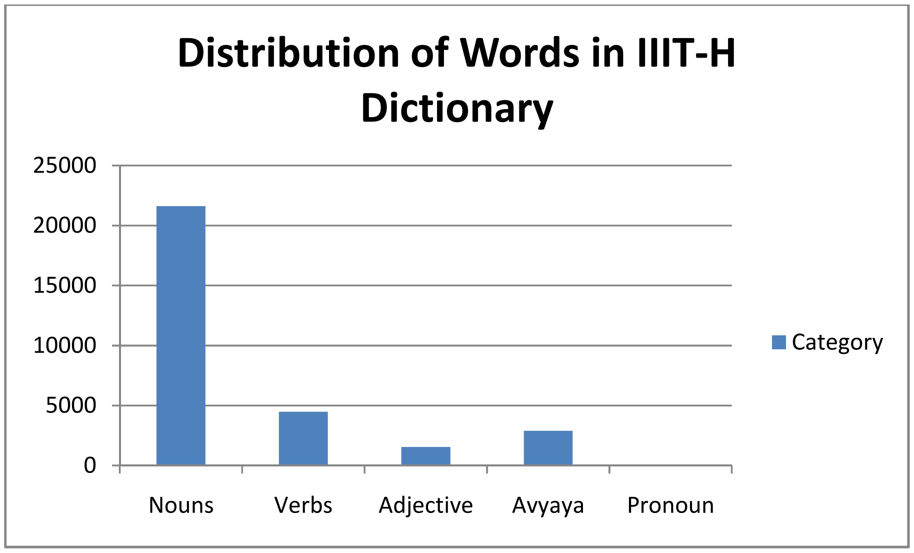


Figure 4.8. IIIT-H Kannada- Hindi Dictionary

In the first stage, we performed coverage analysis of our maindictionary (closed class words with all inflected forms, nouns, adj,adv) with other dictionaries. In below table we have shown number of words by category.

Table 4.9. Closed Class Words Types

Category Description	No. of Words	Category Description	No. of Words
Pronoun	1819	Postposition	120
Basic Adjective	3717	Cardinals	356
Adjective Quantifiers	68	Cardinals human	289
Adjective Ordinals	220	Noun locative Place	470
Adjective Demonstrative	4	Noun Locative Time	658
Adverb of Place	47	Punctuation	18
Adverb intensifiers	22	Postposition	120
Adverb of question	64	Adverb of others	180
Adverb of Duplication	117	Adverb of Negatives	12
Adverb of Manner	368	Conjunctuation	102
Adverb of Time	183	Interjection	65

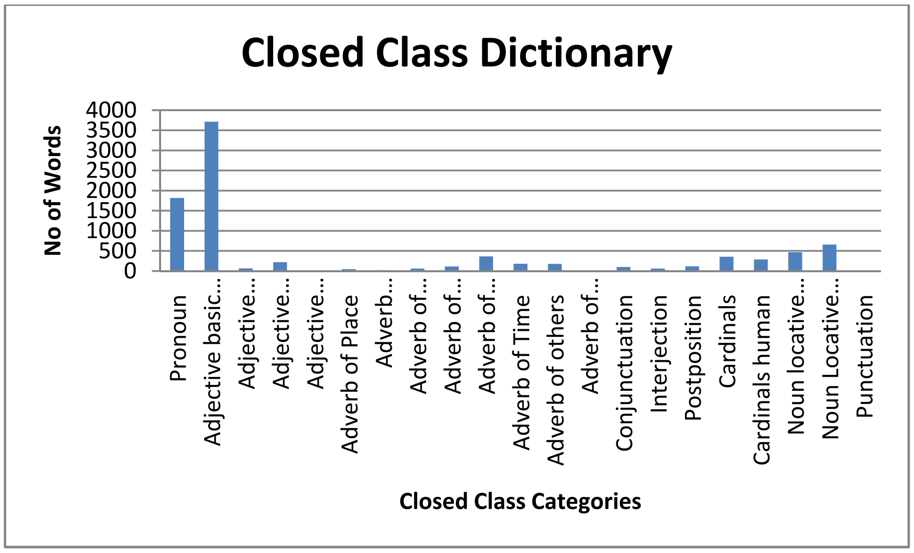


Figure 4.9. Corpus Based Closed Class Dictionary

The observed ambiguity in our dictionary is shown in below table.

Table 4.10. Ambiguous Words in the Morphological Dictionary

Ambiguity Type	**No. of Words**	**Ambiguity Type**	**No. of Words**
ADV-PRO	17	ADV-ADV	8
N-V	231	N-PP	25
V-V	9	ADJ-N-PTN	3
ADJ-N-V	14	ADV-PP	7
N-N	157	ADJ-INTJ	1
ADJ-ADV	24	ADJ-V-V	1
ADV-N	10	PP-V	1
ADJ-N	236	INTJ-V	1
ADJ-PTN	621	ADJ-INTJ-N	1
ADV-VOC	2	ADV-N-PP-V-V	1
N-N-N	2	N-PRO-V	1
ADJ-PP	2	ADV-ADV	8
ADJ-ADV-PP	2	N-PP	25
CONJ-PP	2	ADJ-N-PTN	3

CONJ-N	4	ADV-PP	7
ADV-CONJ	10	ADJ-INTJ	1
ADJ-PRO	1	ADJ-ADJ	1
N-PRO	1	INTJ-N	1
ADJ-N-PP	1	N-PP-V	1
PRO-V	1	ADJ-PRO	1
ADJ-ADJ	1	N-PRO	1
INTJ-N	1		

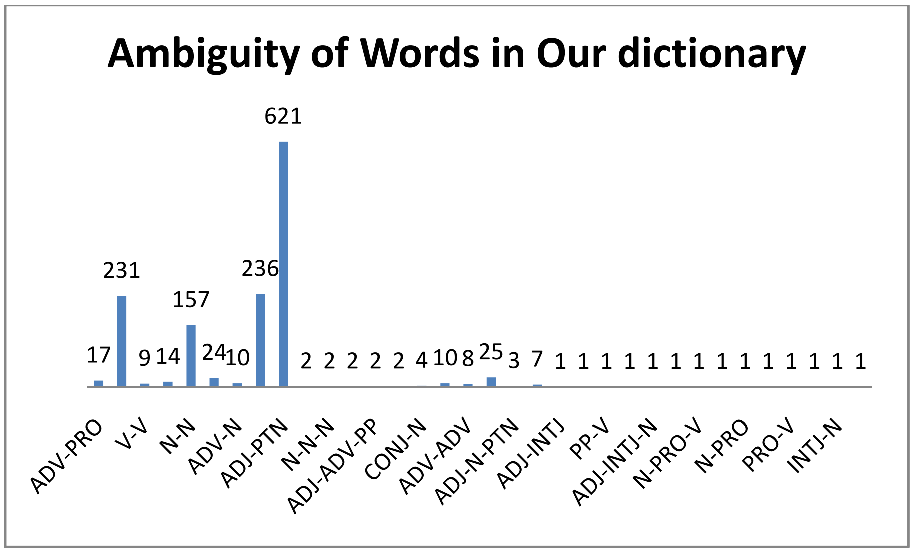


Figure 4.10. Ambiguity observed in our Dictionary

We carried out closed class (CC) words coverage analysis separately on few sample files selected from **DOE CIIL Corpus. The total word list of the corpus is 3,42,535**. The analysis is shown in box 4.4 and box 4.5. Around 3051 of Closed Class words from our dictionary are matched. That is, about 50% of the words in this dictionary are never used. This could be because of infrequent usage of clitics etc. It could also be because of spelling errors.

```
Token-wise Coverage: YOGA.aci:
File has 6783 tokens. 1587 closed class words
detected.
Token-wise Coverage: 23.40 %.
```

Box 4.4. Coverage of Closed Class Words in File YOGA.aci

```
AMERICA1.aci File has 11002 tokens.
1508 closed class words detected.
Token-wise Coverage: 13.7065988002181 %.
```

Box 4.5. Coverage of Closed Class Words on File AMERICA.aci

Our heuristic methods have produced a dictionary of 30000 plus words. It is an ongoing work.

4.8 Applications of an Electronic Dictionary

Electronic dictionaries are directly usable by computer programs. Electronic dictionaries or lexicons, as they are popularly known, in NLP and linguistics, form an integral component of NLP applications.They do almost every activity in computational linguistics and NLP vocabulary studies like spelling error detection and correction, word processing and text critiquing systems, automatic abstracting and indexing, concordances and frequency analysis, and other statistical studies, stylistics, lexicographers' morphological analysis and synthesis, parsing, story understanding, taxonomical studies, knowledge acquisition, linking other systems with knowledge bases, machine translation, question answering systems, natural language interfaces to databases, information retrieval, speech synthesis, speech recognition, computer aided instruction, psycholinguistic studies, office automation, etc. Properly designed electronic dictionaries can be used as a 'plug in modules' by a number of different kinds of NLP applications. Building dictionaries as resources for natural language processing applications is a valuable effort.

4.9 Advantages of Morph Related Dictionary Over Existing dictionaries

Proposed Morph related dictionary has many advantages over existing dictionaries. Advantages of our dictionary with other dictionaries is considered by taking the parameters like size, type of tagging, sub category the gist of comparison is shown in below table 4.11.

Table 4.11. Comparison with Existing dictionaries

Characteristics	**Our Morph Related Dictionary**	**Prof. V.Krishna Dictionary**	**Prof. K N. Murthy Dictionary**	**IIIT-H Kan-HinDictionary**	**Mysore University Dictionary**
Size in Words	31548	1,60,039	10505	30566	1,06,102
Number of Category	10	9	5	5	-
Categorization in adverbs	Yes	No	Limited	No	No
Adverb category	Yes	Yes	No	No	Yes
Sub categorization in adjective	Yes	No	No	No	No
Inflected forms Irregular Morphology	Yes	No	No	No	No
Dictionary Entry	`guru || N-COM-COU-M.SL-NOM::LV-real u`	`aMgacitta (n.)`	`lex($kaTTu$, [[v],[n]]).`	`:"_oMdu","avyaya ","avy","_eka"`	
Tagging	Hierarchical	Flat	Limited	Flat	Flat
Morph related Information	Stored	No	Yes	No	Yes
Pronunciation or the etymological	No	No	No	No	Yes

properties					
Encoding	New Roman	Unicode	Old notation	ISCII	PDF
Source	Developed in this work	Personal communication	Personal communication	URL= http://ltrc..iiit.ac.in /online services/ Dictionaries	CD form
	-Monolingual Dictionary -Developed with Hierarchical tag set	-Monolingual Dictionary -Developed with Flat tag set	-Monolingual Dictionary -Developed with flat tag set	- Bilingual dictionary -Tagged with flat tag set	-Meant for end user to look up meaning of English word in Kannada
Applications	Useful for morphological processor, Chunking, MT	To less extent	To less extent	Designed for MachineTranslation	Available in PDF form
	Useful in parsing	-	-	-	-

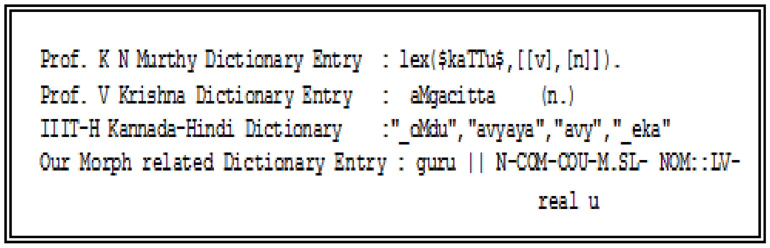
i) We observe that available dictionaries do not give detailed morpho syntactic information and often do not agree with one another even with respect to the main grammatical category. This happens because of the following reasons.

a. Basis principles adopted for categorization could be somewhat different in each case. A noun could act like an adjective and modify another noun and so-on some dictionary may treat it as a noun while the other may treat it as an adjective.

ii) We need not carry the meaning, pronunciation or the etymological properties of lemma etc. Our requirement is that the tag we are assigning tag for the lemma is to be hierarchical to catch more Morphological (descriptive) information about the word. For example if a lemma is a noun, we are giving the full description of word like noun type as a common/proper/locative, whether countable or uncountable, human or non human, nominative case in general.We are assigning a descriptive tag to each entry in lexical entry in the dictionary.

iii) NLP applications like morphological analyzer, parsing, MT (Machine Translation) requires detailed information of the word for better analysis. We have used a hierarchical tagset for tagging words in the dictionary. The hierarchical tagset was developed in first stage of research.

iv) Grammatical information is also stored in the dictionary using the separator **:** (double colon) to help morphological generator system. In Kannada the words that end with 'u' behave differently i.e. if a word ends with real 'u' then glide 'v' is inserted during saMdhi, if the word ends with enunciative 'u' then no glide is inserted during saMdhi and ending 'u' is also deleted. This information is captured in the dictionary and all real 'u' ending words are stored with **extra information indicator field**. The word guru (teacher) is a wordending with **real 'u'**. But the word niiru (water) even though it ends with u it is not real 'u', **it is enunciative**. It is necessary to store such useful information for morphological generator. **Similarly mentioning of countable and uncountable features for nouns is also important,** this feature helps in avoiding the generation of invalid forms. If the word has uncountable feature then such words will not have plural forms like salt, water, gold etc. Sample entry in of all existing dictionaries and proposed dictionary is shown below box 4.6.



```
Prof. K N Murthy Dictionary Entry  : lex($kaTTu$,[[v],[n]]).
Prof. V Krishna Dictionary Entry   :  aMgacitta    (n.)
IIIT-H Kannada-Hindi Dictionary    :"_cMdu","avyaya","avy","_eka"
Our Morph related Dictionary Entry : guru || N-COM-COU-M.SL- NOM::LV-
                                                 real u
```

Box 4.6 : Sample Entry from different Dictionaries

v) Locative nouns are often treated as adverbs of place in a few dictionaries. We need to have a consistent and clearly defined set of major grammatical categories.

Here we use morphological and syntactic consideration sonly. Thus any word which can takes a plural can only be a noun and not an adjective. If a case marker can be added, it can only be a noun ,not an adjective.

vi) We have seen that electronic lexicons are extremely useful both for human users and for NLP programs. We have noted that the current situation in our country has emphasized the need for serious effort towards building electronic dictionaries for Indian languages. We do hope that these discussions would help us in being realistic and in being on the right track in all our endeavors to build electronic lexicons. A broad coverage dictionary is sufficient to get a high rate of accuracies.

vii) A lexical category is a syntactic category for elements that are part of the lexicon of a language. These elements are at the word level. Also known as part of speech word class grammatical category. A lexicon exhibits the knowledge that a native speaker has about a language. We have described our on-going work in developing a large scale dictionary for Kannada including a detailed morpho-syntactic tags. We use the available electronic dictionaries and text corpora and build the dictionary in stages using heuristic pattern matching. Although none of the sources we use are entirely reliable, a large modern monolingual dictionary of any language has an important role to play in the community. Language communities whether large and small need their dictionaries. An essential requirement is a lexical database.

- **Summary**

An electronic dictionary of around 30,000 plus words using semiautomatic methods is developed as a valuable resource in NLP applications. A Closed class dictionary of 8988 words is developed by referring different grammar books as part of the main dictionary. We have manually checked and tagged all words which we got from the intersection of the existing lexicons, since they were tagged with flat tagset and the treatment of words was found inconsistent in the available dictionaries hence lot of preprocessing work was done in order to bring the dictionary in to the current shape. Another important aspect in developing electronic monolingual dictionary is the knowledge of a native language, which helps in designing heuristics and regular expressions for extracting patterns from the corpus. Properly designed electronic dictionaries can be used as 'plug in modules' by a number of different kinds of NLP applications. Electronic dictionaries are assets for computational linguistics.

Chapter 5

Proposed Automatic Morphosyntactic Annotator System Architecture

In chapter 4, we discussed how to develop an electronic dictionary using the available resources within a few months of time and also showed that developing a dictionary using hierarchical tagset is more advantageous than developing using the flat tagset. In this chapter proposed AMAS architecture is explained and development of morphological analyzer/generator (MAG) applying finite state transducers technology is discussed. *MAG component will make use of a hierarchical tagset and the dictionary which we have originally developed (vide chapter 3 & 4).* MAG is the main component of our AMAS architecture. We have developed a wide coverage morphological analyzer/generator to handle most of all valid inflected word forms in Kannada. The morphological module gives morpheme by morpheme output of word formation process. *The proposed MAG system handles inflectional morphology, derivational morphology, clitics aspect, contingent features, formation of new conjunct verbs by the conjugation of aspect verbs to the lexical verbs, and finally external saMdhi.* The results are encouraging for nouns as compared to verbs. *Our morphological analyzer is the first system developed with the hierarchical tagset approach. We observed that the analyzers developed with a flat tagset are not able to capture the deep syntactic information.* Many practical difficulties exist like handling of ambiguity among the words, handling of unknown words that occur in a running corpus, handling of compounds and so on. More than 90% accuracy for nouns and around 85% for verbs and 100% for pronouns is achieved through the proposed morphological analyzer/genearator.

5.1 Overview of Morphology

Morphology is the study of the internal structure and transformational process of word. Words are formed from a combination of one or more free morphemes and zero or more bound morphemes. Free morphemes are units of meaning which can stand on their own as words. Bound morphemes are also units of meaning; however, they cannot occur as words on their own, they can occur only in combination with free morphemes. The English word jumped is comprised of two morphemes, "**jump", and "ed"**. Since 'jump' is an individual unit of meaning which cannot be broken down further into smaller units of meaning, it is a morpheme and, since 'jump' can occur on its own as a word in the language, it is a free morpheme. The unit +ed can be added to a large number of English verbs to create the past tense. Since **+ed** has meaning, and since it cannot be segmented into smaller units, it is a morpheme. However, +ed can only occur as a part of another word, not as a word on its own, therefore, it is a bound morpheme. The process by which bound morphemes are added to free morphemes can often be described, using a word formation rule.

Both analysis and generation rely on two sources of information: A dictionary of valid lemmas of the language and a set of inflections paradigm. The basic principle of morphological generation is to get forms from a root and a set of features (lexical category and morphological properties). Generally, there are two categories of approaches used in developing a morphological generator. Approaches that use finite-state transducers (FSTs), such as Xerox Arabic analyzer (Beesley, 2003) and approaches that use rule based transformations.

The words in a language are finite. But the sentences constructed from these words are infinite and hence all the sentences in a language cannot be stored in the human brain.The association of sound patterns with meanings at sentence level is rule governed. Hence sentences are not stored in human mind. But the association of sound with meaning at word level is not rule governed, hence we need to list all the words in the dictionary or in the mental lexicon. For example, the word 'chair' does not always mean the chair on which one can sit, it means different thing in the context of chair person and so on. The dictionary has to list all possible senses of the word. Words have meanings and meanings are relaed by process of activity, for example the word eat, eating, eats, eaten have eat as their root word. These different forms of eat need not be stored in the dictionary, instead they can be captured by some rules. Similarly words like cat, cats. The study of the structure of words like eat, eating, ate, eaten shows some systematic relationship in the form and function,all these forms are generated from the basic word eat, and it is not necessary to store all these forms in the dictionary. This kind of behavior can be captured by some rules which are known as Morphology. *The change of meaning from 'cat' to 'cats', 'rat' to 'rats' is systematic and productive, it applies to almost every noun, and the productive rules apply to new words also. Similarly eat to eating is also productive and rule governed. Writing of a single rule reduces storing of thousands of words in the dictionary, thereby saving space and time required to search out a word in the dictionary.* Words are the basic building blocks of any language. Words in the language are important. Keeping words i.e. meaning in mind we are proposing a morph-centric architecture for implementing computational grammar for Kannada at word level.

(Mohri, 1997) proposed one of the most efficient approaches to morphological analysis and generation using finite state transducers (FST). There are a number of tools for the construction of FST based morphological analyzers. The best known being those developed at Xerox by (Karttunen,1995) and by (Chanod, 1996), (K Koskenniemi, 1983).

(Beesley, 2003) introduced the first large scale implementation of Arabic morphology within the constraints of finite-state methods. (McCarthy et al., 2002) describes root-and-pattern morphology in the framework of autosegmental phonology has given rise to anumber of computational proposals. (Kay, 1987) proposes a framework in which each of the auto segmental tiers is assigned a tape in a multi-tape finite state machine, with anadditional tape for the surface form. (Kiraz, 1994) extends Kay's approach and implements a small multitape system. (Habash, 2004) proposed a large scale lexeme approach for Arabic morphological generation. Finite state transducers are best suited for implementation of morphological analyzers.

5.2 Finite state transducers

From the Chomsky hierarchy, we can observe that regular grammars are very simple and finite state machines can be used to generate these languages. Finite state technology has some important advantages, making it most appealing for implementing natural language morphology. One can find it very hard, almost impossible, to build the full automaton or transducer describing some morphological phenomena. This difficulty arises from the fact that there are a great number of morpho-phonological processes combining together to create the full language.

However, it is usually very easy to build a finite state machine to describe a specific morphological phenomenon. The class of regular languages is closed under union, concatenation, intersection and complementation. The class of regular relations is closed under union, concatenation and composition (but not under intersection and therefore complementation). These properties make it most convenient to implement each phenomenon independently and combine them together using the closure operations. Moreover, finite state techniques have the advantage of being efficient in their time and space complexity, as the membership problem is solvable in time linear in the length of the input. There are also known algorithms for minimizing and determining automata and some restricted kinds of transducers. All of the above indicate that finite state techniques are a vital tool in the implementation of natural language morphology.

(Alica et al.) defines the transducer as "$T = (Q, L, \delta, qI, F,)$ where Q is a finite set of states, L is a set of transition labels, $qI \in Q$ is the initial state, $F \subseteq Q$ is the set of final states, and $\delta: Q \times L \rightarrow 2^Q$ the transition function (where 2^Q represents the set of all finite sets of states). L is set of transition labels defined as $L = (\Sigma \cup \{\varepsilon\} \text{ x } (\Gamma \cup \{\varepsilon\})$. Input symbols set is Σ, Γ is the alphabet of output symbols and empty symbol is represented as ε. As per this definition, four kinds of state transition may be defined, $(\sigma: \gamma)$, meaning that symbol $\sigma \in \Sigma$ is read and symbol $\gamma \in \Gamma$ is written $(\sigma: \varepsilon)$, meaning that a symbol is read but nothing is written, $(\varepsilon: \gamma)$, means that nothing is read but a symbol is written, and $(\varepsilon: \varepsilon)$ means that a state transition occurs without reading or writing. It is customary to represent the empty symbol ε with a zero ("0"). A letter transducer is said to be deterministic when $\delta: Q \times L \rightarrow Q$. Note that a letter transducer which is deterministic with respect to the alphabet $L = (\Sigma \cup \{\varepsilon\} \text{ x } (\Gamma \cup \{\varepsilon\})$ may still be nondeterministic with respect to the input Σ ".

"A string $w' \in \Gamma^*$ is considered to be a transduction of an input string $w \in \Gamma^*$ if there is at least one path from the initial state qI to the final state in F whose transition labels form the pair $w : w'$ when concatenated. There may in principle be more than one of such paths for a given transduction, this should be avoided, and is partially eliminated by determinization. On the other hand, there may be more than one valid transduction for a string w (in analysis, this would correspond to lexical ambiguity, in generation, this should be avoided). In analysis, the symbols in Σ are those found in texts, and the symbols in Γ are those necessary to form the lemmas and special symbols representing morphological information, such as <noun>, <feminine>, <first Person p1, second person p2, third person p3> for pronouns, etc. In generation, Σ and Γ are exchanged. The general definition of letter transducers is completely parallel to that of non-deterministic finite automata (NFA) and that of deterministic letter transducers parallel to that of DFA; accordingly, letter transducers may be determinate and minimized (with respect to the alphabet L) using the existing algorithms for NFA and DFA (Hopcroft & Ullman 1979; Salomaa 1973; van deSnepscheut 1993). The set of states that can be reached from *p* is called the **epsilon-closure** or **ε-closure** of *p".* The ε -closure of *P* allows a transformation to a new state without consuming any input symbols. For example, if NEA is in state 1, with the next input symbol as *a*, it can move to state 2, without consuming any input symbols also, thus there is an ambiguity, whether the system in state 1, or state 2, before consuming the letter *a*. Because of this ambiguity, it is more convenient to talk of the set of possible states that system may be in. Thus, before consuming letter *a*, the NFA-epsilon may be in any one of the states out of the set {1, 2}. Equivalently, one may imagine that the NFA is in state 1 and 2 'at the same time' and this gives an informal hint of the power set construction 2^Q. The null transitions are allowed since it is a NFA automaton. **We are not preserving any fixed length pattern strings since we perform transitions on suffixes not on single letter.** Separate rule file is used to hold set of orthographic rules, it is not a part of the dictionary. **The idea behind holding a separate file is, to make the system language independent, the code is not hard wired i.e., SaMadhi rules governing insertion or deletion of vowels are put in separate file and not written as part of code.** And another advantage is the same code can be used for other languages too just by replacing orthographic rule file with their language morphophonemic changes. The valid transitions may be an entry in another file called FST transitions file.

5.3 Proposed AMAS Architecture

The architecture of the proposed automatic morpho-syntactic annotator system is shown in figure 5.1. The main aim here is to design a system which takes a Kannada sentence or word as input and assigns POS categories to the input. The design process for the system involves the following tasks: Read the Kannada input which is in Unicode or ISCII convert it into the Roman form using transliterator program then preprocessing is done by passing it to tokenization module, then dictionary lookup is performed then input is passed to morph module and lastly to named entity recognizer module.

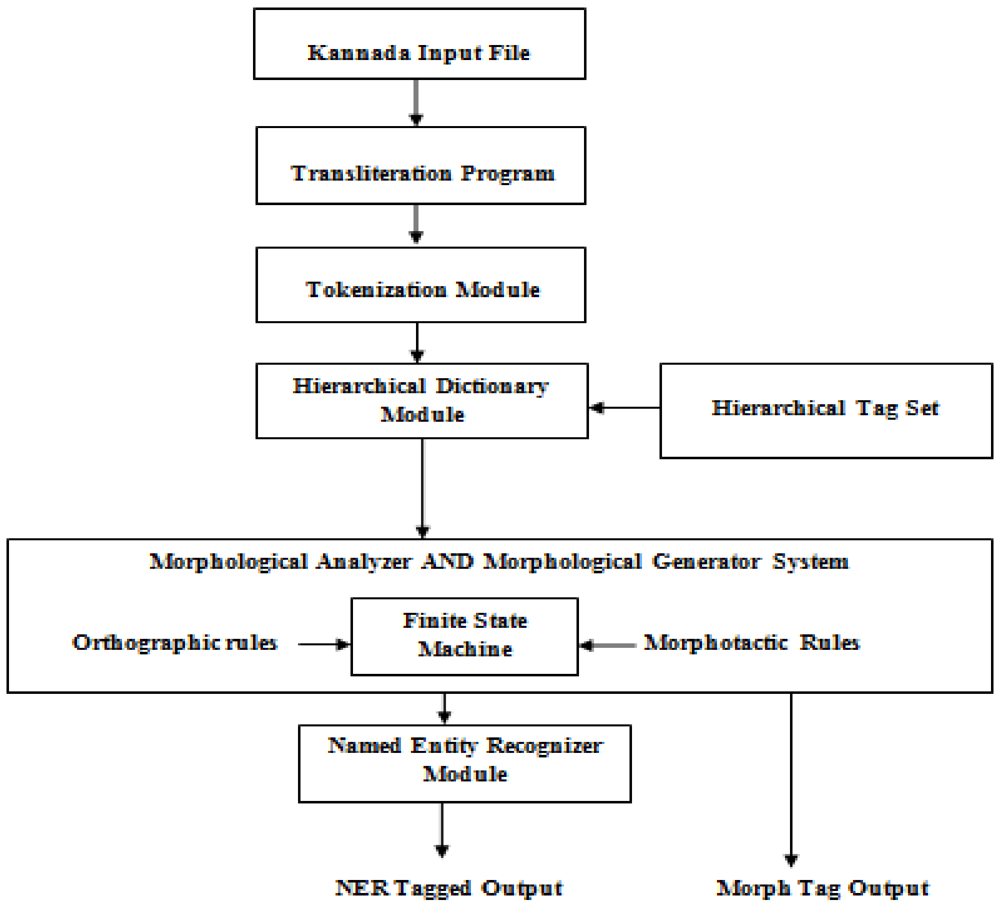


Figure 5.1. Proposed Architecture of AMAS Annotator

5.3.1 Transliteration

First the input is raw corpus which is in UNICODE or ISCII format, it is transliterated in to roman form by using ir.pl (for ISCII) or ur.pl (UNICODE) program. Intermediate map file is used for conversion between Kannada and English characters. After the conversion input is passed to tokenization module for preprocessing.

5.3.2 Tokenization

Tokenization process divides the text file into sequence of tokens, and removes the delimiters like [.!?:;] which are not part of words and prepares the input ready for dictionary lookup process.

5.3.3 Searching Word in Morphological Dictionary

Each token is looked up in the dictionary, **hashing technique of (key, value) pair** is used for searching words in the dictionary. If word is found in the dictionary then corresponding tag is assigned to that token. If the word is not found then it is may be an inflected word or a proper noun, then the word is passed to the morph module to identify its possible inflections. A dictionary of 30,000 words is developed for this work. Dictionary stores morph related information and controls the morphological generator to be over general which otherwise generates many ungrammatical word forms.

The morphological dictionary is a text file. Any text starting with #" is ignored since it is used as comment. The dictionary has the following three sections: i. the symbol declaration section representing the actual root words in the language. 2. the second section represents categories of the word representing representing morphological features like<feminine >, <singular> etc. First and second fields are separated by two vertical bars ||. 3. Section three is the information section, where any morph relevant information like **real u, past participle form of irregular verbs** etc. are declared. While generating the inflections for a word the information field of the dictionary is looked. The development of a dictionary varies with a specific requirement. The purpose of development of dictionary here is we require a dictionary of the type, lemma followed by its tag suitable for morphological analyzer/generator process. The structure of our morphological dictionary entry is shown in box 5.1.

Word	Seperator1	Tag1	Tag2	Seperator2	Morph related Information

Box 5.1. Structure of Dictionary Content

One entry per line. <Word>||<TAG>||<TAG>: Relevant Morph information. Grammatical information useful for our morphological generation system is also stored in the dictionary using the separator**:** (double colon) to help morphological generator. For example consider the word ‘guru’ (teacher), here this word ends with real ‘u’. But the word ‘niiru’ (water) even though it ends with u, it is not real ‘u’, it is enunciative. When words ending with ‘u’ participate in saMdhi, there is insertion of glide ‘v’ for real u cases, for enunciative ‘ u’ there is deletion of ‘u’, such information useful for the morphological generator. We have stored such morph related information in the dictionary as shown here guru|| N-COM-COU-M.SL-NOM-NULL: REALu. Box 5.2 shows the format entries.

```
taavu ||PRO-REF-P23.MFN.PL-NOM
tamma||PRO-REF-P23.MFN.PL-GEN||N-COM-COU-M.SL-NOM::TYPE-kinship
biMdu ||N-COM-COU-N.SL-NOM::LV-real-u
cakshu ||N-COM-COU-N.SL-NOM::LV-real-u
caru ||N-COM-COU-N.SL-NOM::LV-real-u
daaru ||N-COM-UNC-N.SL-NOM::LV-real-u
dattu ||N-COM-COU-N.SL-NOM::LV-real-u
```

Box 5. 2. Sample Entries in the morphological dictionary

5.3.4 Orthographic Rules Set: Morphological analyzers/generators use the concatenation process while they analyze/generate a word-form. So, we must create a set of concatenation rules for the analyzer/generator according to the language. The rules change with word endings as shown in table 5.1. The alternative forms (spelling changes) of morphemes according to the context in which they appear.

Table 5.1: Glide Insertion Based on word Endings

	Stem Ending						
	a	**e**	**i**	**o**	**u**		**consonant**
					Real	**Enunciative**	
Glide insertion for rational	**n**	**y**	**y**	**n**	**v**	**u dropped**	**n**
Glide for non rational	**v**	**y**	**y**	**v**	**v**	**u dropped**	**-**

5.3.5. **Morphotactic Rules**: These rules govern the order of suffixation. Generally the additions of suffixes to noun and verb follow the pattern shown in figure 5.2 below.

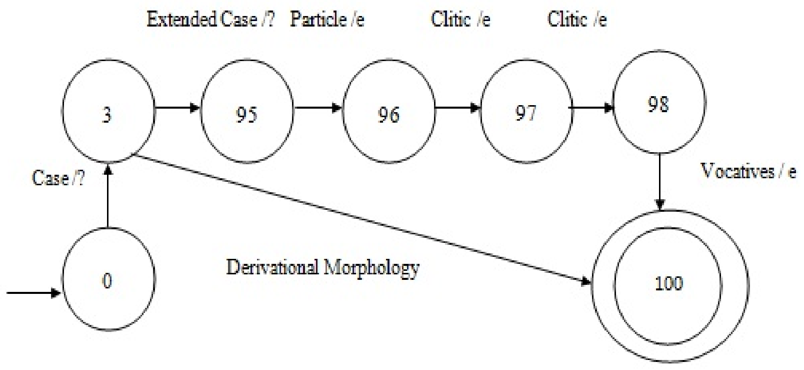


Figure 5.2. Example of Morphotactic Rule for Kannada nouns

5.3.6 Working of the Proposed Mophological analyzer and Generator (MAG) System

Morphological analyzer and morphological generator (MAG) is the main component of our AMAS architecture. The morphological analyzer does analysis of the formation of the complex or inflected word or derived word, where as the morphological generator generates the complex word given the root word and its feature (generally possible inflections/suffixes). The MAG system makes use of orthographic rules and follows the morphotactics. *12 Case or Vibhakti rules, 2 number rules (Singular/Plural), 3 gender rules (masculine, feminie, neuter) are used. 6 Clitic morphology rules are also handled. A Total of around 450 rules are used.* .

Working of FSM: Our FSM always goes from state labeled 0, to state labeled 100. Only state labeled 100 is Final State. We use *phi* (ε) to indicate null or empty suffixes. We use *string or pattern* transducers instead of *letter transducers* unlike (Roche Schabes1997). Any finite-state transducer may always be turned into an equivalent letter transducer. Instead of transition on letters, here *transitions are on sequences of letters i.e., strings*, which are generally, valid suffixes in the language. The machine starts in the specified initial state and reads a string or alphabet, the automaton uses the state transition function to determine the next state using the current state, based on the symbol just read or the empty string. However, "the next state of an NFA depends not only on the current input event, but also on an arbitrary number of subsequent input events". Until these subsequent events occur it is not possible to determine which state the machine is in. When the automaton has finished reading, it is in an accepting state, the NFA is said to accept the string, otherwise it is said to reject the string. When the last input symbol is consumed, the NFA accepts if and only if there is some set of transitions that will take it to an accepting state. It is non-deterministic meaning, for any input symbol, its next state may be any one of several possible states. Thus, in the formal definition, the next state is an element of the power set of states. Category wise set of rules are compiled into subtransducers that are then integrated to build the complete transducer. **There is peculiar behavior of a suffix 'a' in Kannada. i.e. the transitionon suffix 'a' in Kannada stands for a genitive case of nouns , and also for negation for verbs, and also for another sense of imprecate meaning of verb**. In such ambiguiguos case Morphological analyzer gives more than one analyses. All are valid transitions, since context is not considered here. Many challenges were imposed during the design of Morphological analyzer and generator. **The real complexity of the language is studied only when such languages are explored computationally**. Sample of limited version of FSM transitions is shown in figure 5.3 for verb root "maaDu" (do).

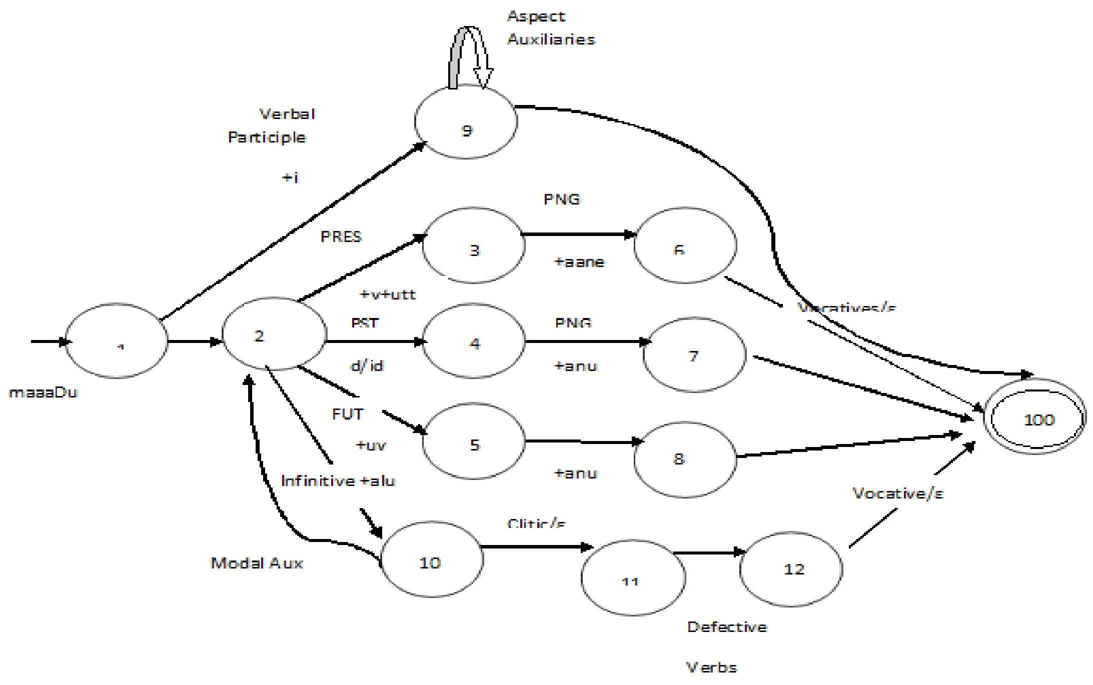


Figure 5.3. Sample FSM Transition for Verb: A limited version

This is the first morphological analyzer which handled, most of all the morphological aspects of Kannada covering inflectional, derivational, clitics, voice modifiers, modal auxiliaries, aspect auxiliaries etc. All possible inflectional are listed in table 5.2. The following table shows an exhaustive list of the various transition suffixes, and states in inflectional and derivational Morphology from verb to noun, verb to adjective and vice versa. (v-verb, n-noun, adj-adjective).

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
# FSM for noun and verb morphology of Kannada
# START_STATE END_STATE n/v SUFFIX FEATURE_VALUE+

# FSM: Always goes from state labeled 0 to state labeled 100.
# only state labeled 100 is Final State

# Use phi to indicate null or empty suffixes
# These are abstract suffixes - may be replaced using suffix-map
table

#################################################################
######
# Noun Morphology

# Number

0     3     n     gaLu       plural
0     3     n     phi        singular

# Case:

3     95    n     annu       accusative
3     95    n     iMda       ablative
3     95    n     ige        dative
3     95    n     alli       locative

3     100   n     a          genitive
3     92    n     a          oblique
3     95    n     (u)        nominative
```

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
3     100   n     uu          clitic_uu
3     100   n     aMtuu       clitic_aMtuu
3     100   n     ellaa       clitic_ellaa

# Case for Derived Nouns: Ex. maaDuvudu

4     95    n     annu        accusative
4     95    n     ariMda      ablative
4     95    n     akke        dative
4     95    n     aralli      locative

4     100   n     ara         genitive
4     92    n     a           oblique
4     95    n     (u)         nominative

4     95    n     aroDane     sociative_1
4     95    n     aroMdige    sociative_2
4     95    n     aroTTige    sociative_3
4     95    n     akkiMta     comparative
4     95    n     aroLage     locative_2
4     95    n     arallige    locative_dative_1
4     95    n     aroLakke    locative_dative_2

4     95    n     akkooskara  purposive_1
4     95    n     akkaagi     purposive_2
4     95    n     aramuulaka  via
4     95    n     oMdu        oMdu
4     95    n     aaru        yaaru_interrogative

4     95    n     illa        existential_negative

4     100   n     uu          clitic_uu
4     100   n     aMtuu       clitic_aMtuu
4     100   n     ellaa       clitic_ellaa
```

Table 5.2 State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
# Case-Like Suffixes:

3     95    n     oDane       sociative_1
3     95    n     oMdige      sociative_2
3     95    n     oTTige      sociative_3
3     95    n     igiMta      comparative
3     95    n     oLage       locative_2
3     95    n     allige      locative_dative_1
3     95    n     oLakke      locative_dative_2
3     95    n     igooskara   purposive_1
3     95    n     igaagi      purposive_2
3     95    n     illa        existential_negative

# Derivational Morphology:

3     100   n>adv aMte        similaritive_1
3     100   n>adj aMtaha      similaritive_2
3     100   n>adj aMtha       similaritive_2

3     100   n>adj allada      negative_rp
4     100   n>adj allada      negative_rp

##################################################################
######
# Verb Morphology

0     100   v     a           imprecative

0     15    v     isu         causative
0     15    v     phi         null

0     98    v     isu         causative_imperative
```

Table 5.2 State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
# Aspect

15    17    v     uttiruttiru iterative
15    17    v     uttiru      progressive

15    17    v     phi         null

15    98    v     uttiruttiru iterative_imperative
15    98    v     uttiru      progressive_imperative
15    98    v     iru         perfective_imperative

# 17 is final state - singular imperative

# Tense
# non_past includes present/future/habitual/generic

# future is used only in formal usage

17    20    v     id          past
17    21    v     utt         non_past
17    22    v     uv          future

# Special Forms: Combining Tense and Aspect: ref. iddaane
# maaDiddane - present perfective
# maaDuttidaane - present progressive
# maaDiruttane - used in future sense
# maaDuttiruttane - used in future sense

15    23    v     uttidd            progressive_present

# maaDide, maaDive:
15    95    v     uttide      progressive_present_p3_n_singular
15    95    v     uttive      progressive_present_p3_n_plural

15    95    v     uttilla     progressive_present_negative
```

Table 5.2 State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
# maaDikoMDide
30    95    v     ide         perfective_present_p3_n_singular
30    95    v     ive         perfective_present_p3_n_plural

30    95    v     illa        perfective_present_negative

# Finite Forms

20    95    v     e(nu)       p1_mfn_singular
20    95    v     evu         p1_mfn_plural
20    95    v     e           p2_mfn_singular
20    95    v     iri         p2_mfn_plural
20    95    v     anu         p3_m_singular
20    95    v     aLu         p3_f_singular
20    95    v     aru         p3_mf_plural
20    95    v     itu         p3_n_singular

20    95    v     avu         p3_n_plural

21    95    v     eene        p1_mfn_singular
21    95    v     eeve        p1_mfn_plural
21    95    v     ii(ye)      p2_mfn_singular
21    95    v     iiri        p2_mfn_plural
21    95    v     aane        p3_m_singular
21    95    v     aaLe        p3_f_singular
21    95    v     aare        p3_mf_plural
21    95    v     ade         p3_n_singular
21    95    v     ave         p3_n_plural
22    95    v     e(nu)       p1_mfn_singular
22    95    v     evu         p1_mfn_plural
22    95    v     e           p2_mfn_singular
22    95    v     iri         p2_mfn_plural
22    95    v     anu         p3_m_singular
22    95    v     aLu         p3_f_singular
```

Table 5.2 State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
22    95    v     aru         p3_mf_plural
22    95    v     adu         p3_n_singular
22    95    v     avu         p3_n_plural
23    95    v     eene        p1_mfn_singular
23    95    v     eeve        p1_mfn_plural
23    95    v     ii(ye)      p2_mfn_singular
23    95    v     iiri        p2_mfn_plural
23    95    v     aane        p3_m_singular
23    95    v     aaLe        p3_f_singular
23    95    v     aare        p3_mf_plural
17    100   v     ali         optative
17    100   v     alii        optative_conjunctive
17    100   v     alaa        optative_interrogative
17    100   v     aloo        optative_interrogative
17    100   v     alee        optative_interrogative
17    95    v     ooNa        hortative_p1_plural
17    100   v     i(ri)       plural_imperative

17    100   v     iree        plural_imperative_vocative_feminine
17    100   v     iroo        plural_imperative_vocative_masculine
17    100   v     (u)         singular_imperative

# Required to Handle maaDatoDagu, maaDahattu etc.
# To check the effect of this arc on over generation

17    95    v     iddilla     past_negative

17    95    v     uvadilla    future_negative
# adu

15    95    v     enu         future_negative_p1_mfn_singular
15    95    v     evu         future_negative_p1_mfn_plural
15    95    v     e           future_negative_p2_mfn_singular
15    95    v     ari         future_negative_p2_mfn_plural
15    95    v     anu         future_negative_p3_m_singular
```

Table 5.2 State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
15    95    v     aLu         future_negative_p3_f_singular
15    95    v     aru         future_negative_p3_mf_plural
15    95    v     adu         future_negative_p3_n_singular
15    95    v     avu         future_negative_p3_n_plural

# Non-Finite Forms - Past

17    100   v     e           conditional

20    93    v     are         conditional
20    93    v     oDe         conditional

20    100   v     aruu        concessive

# Non-Finite Forms - Non_Past


21      100     v       aa        cjp_present
21      100     v       alee      cjp_present_emphatic
21      100     v       aluu      cjp_present_inclusive
21      100     v       aloo      cjp_present_indefinite

# Non-Finite Forms - Future

# Infinitive

17    25    v     al(u)       infinitive
17    95    v     alu         infinitive

25    26    v     ee          emphatic
25    26    v     uu          inclusive
25    95    v     illa        past_negative
25    26    v     phi         null

26    28    v     beeku       compulsive
26    95    v     beeku       compulsive
```

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

26	98	v	beeDa	negative_imperative
26	98	v	beeDi(ri)	negative_plural_imperative
26	100	v	beeDavaa	negative_compulsive_interrogative
26	100	v	beeDavoo	negative_compulsive_interrogative
26	100	v	beeDavee	negative_compulsive_interrogative
26	100	v	beeDiroo	negative_compulsive_vocative_masucline
26	100	v	beeDiree	negative_compulsive_vocative_feminine
26	28	v	bahudu	probabilitive_or_permissive
26	95	v	bahudu	probabilitive_or_permissive
26	95	v	takkaddu	compulsive
26	28	v	takkaddu	compulsive
26	28	v	baaradu	prohibitive
26	95	v	baaradu	prohibitive
26	28	v	kuuDadu	prohibitive
26	95	v	kuuDadu	prohibitive
26	15	v	paDu	passive
26	15	v	toDagu	asp_aux_toDagu
26	15	v	hattu	asp_aux_hattu
26	15	v	baru	asp_aux_baru
26	15	v	aaraMbhisu	asp_aux_aaraMbhisu
26	15	v	horaDu	asp_aux_horaDu
26	15	v	bayasu	asp_aux_bayasu
26	15	v	icchisu	asp_aux_icchisu
26	15	v	eLasu	asp_aux_eLasu
26	15	v	etnisu	asp_aux_yatnisu
26	15	v	hoogu	asp_aux_hoogu
26	15	v	iru	asp_aux_iru
26	15	v	aagu	asp_aux_aagu
26	22	v	ball(a)	capabilitative

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

26	4	v>n	balladdu	gerund_capabilitative
26	22	v	aara	non_capabilitative
26	95	v	aara	non_capabilitative_p3_m_singular
26	23	v	idd	perfective_present
28	15	v	aagu	asp_aux_aagu
28	15	v	iru	asp_aux_iru
# Conjunctive (Past Verbal) Participle				
15	30	v	i_u	cjp_past
15	95	v	i_u	cjp_past
30	17	v	iru	perfective
30	15	v	koLLu	reflexive
30	15	v	haaku	asp_aux_haaku
30	15	v	biDu	asp_aux_biDu
30	15	v	aaDu	asp_aux_aaDu
30	15	v	nooDu	asp_aux_nooDu
30	15	v	hoogu	asp_aux_hoogu
30	15	v	koDu	asp_aux_koDu
30	15	v	baru	asp_aux_baru
30	15	v	iDu	asp_aux_iDu
30	15	v	aagu	asp_aux_aagu

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
30    23    v    idd        perfective_present
30    98    v    koLLu      reflexive_imperative
30    98    v    haaku      asp_aux_haaku_imperative
30    98    v    biDu       asp_aux_biDu_imperative
30    98    v    aaDu       asp_aux_aaDu_imperative
30    98    v    nooDu      asp_aux_nooDu_imperative
30    98    v    hoogu      asp_aux_hoogu_imperative
30    98    v    koDu       asp_aux_koDu_imperative
30    98    v    iDu        asp_aux_iDu_imperative
30    95    v    illa       existential_negative

30    100   v    uu         clitic_uu

# Contingent

30    95    v    (y)eenu    contingent_p1_mfn_singular
30    95    v    (y)eevu    contingent_p1_mfn_plural
30    95    v    iiye       contingent_p2_mfn_singular
30    95    v    iiri       contingent_p2_mfn_plural
30    95    v    (y)aanu    contingent_p3_m_singular
30    95    v    (y)aaLu    contingent_p3_f_singular
30    95    v    (y)aaru    contingent_p3_mf_plural
30    95    v    iitu       contingent_p3_n_singular
30    95    v    (y)aavu    contingent_p3_n_plural

# Conjunctive (Past Verbal) Participle

17    35    v    a          negative

35    30    v    de         cjp
35    100   v    de         cjp

###################################################################
######
# Derivational Morphology:
```

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
26    96    v>n   oosuga           oosuga_dative_singular_null

# Gerund: (Derivation v->n)

# gerund = tense + udu, negative is like tense too:

22    4     v>n   udu         gerund
20    4     v>n   udu         gerund
35    4     v>n   udu         gerund

# Nominalizers:
17    3     v>n   vike        v-to-n

# Relative Participles:

# Adjective has no morphology as such. Derived adjectives behave

# like nouns, morphologicaly.

20    50      v>adj   (a)          rp

22    52      v>adj   (a)          rp

26    4     v>n   takkaddu    compulsive_gerund
26    50    v>adj takk(a)     rp_compulsive

26    50    v>adj aarad(a)    rp_non_capabilitative
26    4     v>n   aaraddu     gerund_non_capabilitative

35    50    v>adj d(a)        rp

30    50    v>adj illada      rp_negative
30    4     v>n   illaddu     gerund_negative
30    100   v     illade      cjp_negative

35    4     v>n   baaraddu    baaradu_gerund
```

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

35	50	v>adj	baarad(a)	rp_baaradu
35	4	v>n	lolladdu	olla_gerund
35	50	v>adj	loll(a)	rp_olla
35	50	v>adj	lollada	rp_ollada
35	100	v	lollade	cjp_ollade
35	51	v>adj	loll(a)	rp_olla
50	94	adj>adv	aaga	particle_aaga
50	94	adj>adv	aagalee	particle_aagalee
50	94	adj>adv	aagina	particle_aagina
50	94	adj>adv	aaginiMda	particle_aaginiMda
50	94	adj>adv	meele	particle_meele
50	94	adj>adv	naMtara	particle_naMtara
50	94	adj>adv	baLika	particle_baLika
50	94	adj>adv	kuuDalee	particle_kuuDalee
50	94	adj>adv	haage	particle_haage
50	94	adj>adv	oDane	particle_oDane
50	94	adj>adv	atta	particle_atta
50	94	adj>adv	aMte	particle_aMte
50	4	adj>n	ashTu	particle_ashTu
50	3	adj>n	alli	particle_alli
50	0	adj>n	eDe	particle_eDe
52	94	adj>adv	aaga	particle_aaga
52	94	adj>adv	aagalee	particle_aagalee
50	94	adj>adv	aagina	particle_aagina
50	94	adj>adv	aaginiMda	particle_aaginiMda
52	94	adj>adv	varege	particle_varege
52	94	adj>adv	haage	particle_haage
52	94	adj>adv	tanaka	particle_tanaka
52	94	adj>adv	aMte	particle_aMte
52	94	adj>adv	maTTige	particle_maTTige
52	94	adj>adv	atta	particle_atta
52	100	adj>adv	varegina	particle_varege
52	4	adj>n	ashTu	particle_ashTu
52	3	adj>n	alli	particle_alli
52	0	adj>n	eDe	particle_eDe

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
# Adj+Pron=Pron
50    100   adj   aMtaha     similaritive
50    100   adj   aMtha      similaritive
50    100   adj   phi        null
50    53    adj   aMtaha     similaritive
50    3     adj>n aMtahadu   similaritive_pr_adu
50    3     adj>n aMtahavu   similaritive_pr_avu
50    53    adj   aMtha      similaritive
50    4     adj>n aMthadu    similaritive_pr_adu
50    4     adj>n aMthavu    similaritive_pr_avu
50    53    adj   phi        null
52    100   adj   aMtaha     similaritive
52    100   adj   aMtha      similaritive
52    100   adj   phi        null
52    53    adj   aMtaha     similaritive
52    4     adj>n aMtahadu   similaritive_pr_adu
52    4     adj>n aMtahavu   similaritive_pr_avu
52    53    adj   aMtha      similaritive
52    53    adj   phi        null
51    95    adj>v enu        p1_mf_singular
51    95    adj>v evu        p1_mf_plural
51    95    adj>v e          p2_mf_singular
51    95    adj>v iri        p2_mf_plural
51    95    adj>v nu         p3_m_singular
51    95    adj>v Lu         p3_f_singular
51    95    adj>v ru         p3_mf_plural
51    95    adj>v du         p3_n_singular
51    95    adj>v vu         p3_n_plural
# If we say ava(nu), many wrong anal such as avu+a are coming
53    3     adj>n avanu      pr_avanu
53    3     adj>n avanobba(nu)     pr_avanu_obbanu
```

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
53   3    adj>n avanaata(nu)     pr_avanu_aatanu
53   3    adj>n avaniita(nu)     pr_avanu_iitanu
53   3    adj>n aata(nu)         pr_aatanu
53   3    adj>n aake(yu)         pr_aakeyu
53   100  adj>v avanillade       pr_avanu_illade_cjp
53   3    adj>n avaLu            pr_avaLu
53   3    adj>n avaLobbaLu       pr_avaLu_obbaLu
53   3    adj>n avaLaake(yu)     pr_avaLu_aakeyu
53   3    adj>n avaLiike(yu)     pr_avaLu_iikeyu
53   100  adj>v avaLillade       pr_avaLu_illade_cjp
53   3    adj>n avaru            pr_avaru
53   3    adj>n avarobbaru       pr_avaru_obbaru
53   3    adj>n avarellaru       pr_avaru_ellaru
53   3    adj>n avaraaru         pr_avaru_yaaru_interrogative
53   100  adj>v avarillade       pr_avaru_illade_cjp
53   3    adj>n avugaLu          pr_avu
# Note: maaDabeekaadavugaLannu, avugaLige, avugaLa etc.
#################################################################
######
# Clitics and Particles: Applicable to all categories
92   100  n       varege         particle_varege
92   100  n       atta           particle_atta
92   100  n       aMte           particle_aMte
92   100  n       tanaka         particle_tanaka
92   100  n       maTTige        particle_maTTige
92   100  n       ashTu          particle_ashTu
92   100  n       ashTuu         particle_ashTuu
92   100  n       ashTe          particle_ashTe
92   100  n       ashTee         particle_ashTee
93   100  v       eenu           particle_eenu
# can be question. maaDidareenu biTTareenu - does not matter
93   100  v       allavee        particle_allavee
93   100  v       ashTee         particle_ashTee
93   100  v       allavee        particle_allavee
93   100  v       eenaa          particle_ee_plus_(n)aa
93   100  v       phi            null
```

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

```
94    97    any     ee              clitic_ee
94    97    any     uu              clitic_uu
94    97    any     phi             null
95    96    any     aMte            clitic_hear_say
95    96    any     ashTe           clitic_ashTe
95    96    any     alla            clitic_logical_negative
95    96    any     eeke            clitic_eeke
95    96    any     eMtu            clitic_eMtu
95    96    any     eenu            clitic_eenu
95    96    any     ellaa           clitic_ellaa
95    96    any     allade          clitic_allade
95    96    any     phi             null
96    97    any     aa              clitic_aa
96    97    any     ee              clitic_ee
96    97    any     oo              clitic_oo
96    97    any     phi             null
97    98    any     naa             clitic_interrogative
97    98    any     phi             null
# Vocatives: Applicable to all categories.
98    100   any   appa(a)     vocative
98    100   any   amma(a)     vocative
98    100   any   ayya(a)     vocative
98    100   any   phi         null
##################################################################
######
# External saMdhi: Selected verbs can follow a complete verb form
100   0     v     VERB        +v_saMdhi
100   0     n>v   uMTu        +v_saMdhi
100   0     n>v   aagu        +v_saMdhi
100   0     n>v   iru         +v_saMdhi
100   0     adj>v aagu        +v_saMdhi
100   0     adj>v iru         +v_saMdhi
100   0     adv>v aagu        +v_saMdhi
100   0     adv>v iru         +v_saMdhi
```

Table 5.2. State Transitions for Inflection and Derivational Rules in Kannada Morphology

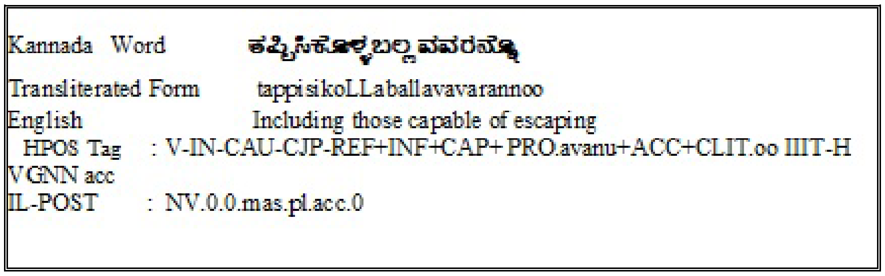
```
# External saMdhi with sub_conj etc:
98   100  any  aaddariMda  +sub_conj_aaddariMda_saMdhi
98   100  any  eMba        +sub_conj_eMba_saMdhi
98   100  any  eMdu        +sub_conj_eMdu_saMdhi
98   100  any  eMduu       +sub_conj_eMduu_saMdhi
98   4    v>n  eMbudu      +sub_conj_gerund_eMbudu_saMdhi
98   100  any  CC_WORD     +cc_saMdhi # Any closed-class word
```

5.3.7 Named Entity Recognizer

The NER system is developed in order to handle proper nouns in the corpus like person names, location names, organization name, If the MAG does not analyze the word, then it may be named entity, so it has to be passed to the named entity recognizer and NER tag will be assigned to the word. Proper noun dictionary of 5,000 words is separately developed for this task. The working of NER system using rule based approach is explained in next chapter 6, two other machine learning approaches Hidden Markov Model and Conditional Random Field are implemented in chapter 7, 8 respectively.

5.4 Kannada Morphology

Kannada is Dravidian language, spoken in southern India. Kannada has complex morphology. Words are built up from roots by following fixed patterns that add prefixes, suffixes to the word. This system that studies how words are constructed from roots, and describes the patterns they follow. Verb formation in Kannada is complicated. For the verb “exists” (iru) we have around 3,000 verb forms. We can add any number of affixes to form a complex verb root. A single root word in Kannada can give rise to a very large number of its surface forms. The richness of Kannada lies in the fact, that significant part of grammar is handled by word morphology. Consider the Kannada verb form shown in box 5.3.

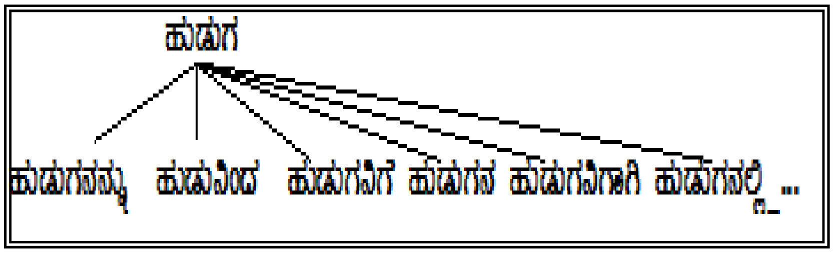

Kannada Word ತಪ್ಪಿಸಿಕೊಳ್ಳಬಲ್ಲವವರನ್ನೂ
Transliterated Form tappisikoLLaballavavarannoo
English Including those capable of escaping
HPOS Tag : V-IN-CAU-CJP-REF+INF+CAP+PRO.avanu+ACC+CLIT.oo IIIT-H
VGNN acc
IL-POST : NV.0.0.mas.pl.acc.0

Box 5.3. Complex Kannada Gerund

This Kannada word form is derived by adding 9 suffixes to the root verb "tappu" (mistake, 'miss'). The formation of this verb form is Verb+-intransitive+causative+Verbal participle form+reflexive + infinitive +modal (CAP) +PRO.avaru+accusative+clitic indefinite.

5.4.1. Noun Morphology

Nouns in Kannada may be distinguished for gender, number, case and clitcs. The following table 5.2 illustrates the various suffixes identified for noun morphology for the computational model. Various Case suffixes, clitics, are identified based on the role they are playing in the sentences and they are named accordingly. A noun'boy' in English has at the maximum 2 inflections boy and boys. But the same word in Kannada has around 250 forms. All these morphosyntactic aspects of the complex morphology have to taken into account for generation. Hence designing morphological generator or analyzer for Kannada language is quite challenging. Few forms out of the 250 are shown in box 5.4



Box 5.4. Different noun Forms of Kannada noun huDuga

- **Gender:**

Nouns referring to biologically female beings are feminine in gender, nouns that are biologically male are masculine in gender and nouns that are not capable thought to be “irrational’ and are referred as neuter. Sometimes young children, small kids are treated as non-rational.

- **Number:**

Kannada nouns are distinguished by two numbers, singular and plural. The singular has no particular distinguishing marker added. The plural marker ’gaLu’ is used for neuter nouns; ’ru’ is used for others. There is an exception in some case where gender is not specific in human noun ’gaLu’ is also used as plural marker, in examples like prajegaLu ’citizen. Usually all masculine and feminine nouns ending in ’a’,’i’, ’e’, and consonant followed by enunciate u have plural marker as ’ru’. Some masculine and feminine nouns are not specially marked for gender like vyaapaari (merchant), adhikaari(officer), these words take plural marker ’gaLu’ which is generally used as neuter plural marker. We are making such words with tag MF(M-masculine, F-feminine), to indicate dual sense, since such words are used to represent both. Some nouns have irregular plurals like makkaLu **‘children’** some nouns have no singular forms, they have only plural usage, like Jana **‘people’**. There are two plurals like janaru (people), janagaLu (people) are two plurals. **aneekaru** means ‘many people’. Some nouns have no usage in plural form; say **gurugaLu ’teacher’** has no corresponding plural.

- **Case system:**

The case system is useful to indicate different relationships between the noun and other constituents of the sentence: Say, to indicate whether the noun is ***object*** of a verb (in which case it is marked for accusative case), or “goal” of a verb of motion (dative case), possessors of something (genitive case), or the means by which something takes place (instrumental case), etc.

- **Nominative case:** The subject always occur in nominative case; for example, raamanu haNNannu tiMdanu.’raama ate the fruit’. raamanu is the subject, haNNannu is the direct object. raamanu is in nominative case and is the subject of the sentence, when we put question on ate, yaaru tiMdaru, and the answer is raamanu, when we ask what did he eat, eenu tiMdanu haNNuannu is the answer and it is the object.

- **Accusative case** is used for transitive/bi-transitive verb. Here the object is also in nominative case. This is allowed in Kannada. The Subject has a role is always in nominative case. But sometimes in dative case alos allowed. For examples ನನಗೆ ಎರಡು ಕಣ್ಣು ಇವೆ. " nanage ereDu kaNNu ive". (I have two eyes) this is special case, here nanage is the subject and it is in dative case, kaNNu is object here which is nominative form. We observed that there is ambiguity between nominative and accusative and also between nominative and genitive cases. However we are not focusing on these issues here.

- **Dative case**: is used for bi-transitive verbs only. Dative has a recipient. Consider the sentence like ರಾಮನು ರಾಧೆಗೆ ಪೆನ್ನು ಕೊಟ್ಟನು. " raamanu raadhege pennu koTTanu". ('raama gave a pen to raadha'). Here raamanu is the subject. raadhege is the indirect object, which is in dative case is receiver, to whom, (yaarige). Direct object is pennu.

- **Ablative / instrumental case:** In Kannada the suffix marker used for instrumental case is "deseyiMda" which means from or by **in** English. The suffix masker for instrumental case is iMda, which means from or by in English. Hence we have considered only one case the ablative. Say ನಾನು ಬಸ್ಸಿನಿಂದ ಬಂದೆನು.naanu **bassiniMda b**aMdenu (I Came by Bus). 'From my place' mainly used in the sense of detaching, going from one place to other.

- **Genitive** case is like possessive in English i.e. x posses y.

Apart from 6 original cases we observe that there are some suffixes which get added to the oblique form of a noun and works like a case. In Kannada the noun gets inflected in the order of case followed by clitics, noun+case+clitics. But not in the form of noun+case1+case2+clitics, for case like suffix. Consider the example below in box 5.5.

Kannada Word Form : ಹುಡುಗನಿಗೋಸ್ಕರ = ಹುಡುಗನ +ಇಗೆ+ ಓಸ್ಕರ
Transliteration : huDuganigooskara = huDugana+ige +ooskara.

Box 5.5 An Example for Purposive Case

Dative case is followed by purposive is not correct. So, we can say that huDugana+igooskara is the correct form treating 'igooskara' as a suffix for Purposive case. The other case-like suffixes are also treated the same way. Table 5.3 gives the details of extended cases and the suffixes. When saMdhi is taking place with oblique and case like forms all e/i/u/a endings are dropped and there is no glide increment added. Case like suffixes are joined with verb oblique forms also, for example *maaDida+oDane = maaDidoDane.* Locative-Dative. But some words like maneyolage, maneyoLakke has the meaning of locative but with a dative suffix, inspite of dative suffix, there is no concept of receiver as in dative case "raamanige siitege" etc. So we have Locative-dative as one more case. Cases and their suffix markers are listed in table 5.3.

Table 5.3. Morphology of Noun Cases and their Characteristics

Case/vibhakti	Suffix/pratyaya	Characteristics	
		Singular Marker	**Plural**
Nominative (prathama)	U	-	ru/gaLu/vu/aMdiru
Accusative (dvitiya)	Annu	nannu/Lannu/vannu/annu/yannu	rannu/aMdirannu/gaLannu/yaranau
Ablative/Instrumental (tritiya)	iMda	niMda/diMda/yiMda/viniMda/LiMda/	gaLiMda/riMda/yariMda/
Dative (Caturthi)	ge/ige/kke	nige/Lige/vige/yige	gaLige/rige/ge
Genetive (shashTi)	A	La/na/da/vina	ra/ gaLa/aMdira
Locative (saptami)	Alli	nalli/yalli/Lalli/dalli/vinalli	ralli/yaralli/gaLalli

Apart from common cases supported, we observed that there are some additional suffixes which also behave like cases. In Kannada we have 6 cases, Nominative, accusative, ablative, Dative, genitive,locative. Additional cases are shown in table 5.4

Table 5.4. Morphology of Extended Noun Case System

Case	Suffix
Purposive	Gooskara/gaagi
Comparative	giMta/kkiMta
Locative 1	Allege
Locative 2	oLage
Locative_Dative1	oLakke
Similarative	aMte/aMtaha/aMtha
Sociative1	oDane/oMdige
Sociative 2	oMdige
Sociative 3	oTTige

5.4.2 Verbs Morphology

Verbs are one of the most interesting and distinguishing aspects of Kannada. Verbs are inflected for tense, aspect, modality. The verb stems in Kannada can be divided into finite and nonfinite. A finite verb ends the sentence with the exception of clitics. But the nonfinite verbs can not stand alone and requires finite verb for ending the sentences. These include infinitives modal auxiliaries, aspect auxiliaries. Finite verbs include imperatives.

DERIVATIVE STEMS: These are formed by adding various kinds of derivative suffixes to the verbal or nonverbal stem. Consider the following examples.
Derived Stem (1) tinnu 'eat'+/alu/→ tinnalu, (to eat) tinnu is the basic stem.

COMPLEX VERBS: These are formed by adding various kinds of models to theprimitive and derived stems. These can be further subdivided into compound verbs, conjunct verbs, modal verbs and aspectual verbs. Kannada has four nonfinite forms of verbs. Like the participial constructions, verbal participles,relative participles,they are aspectualas distinguished from general morph inflections. They are shown in table 5.5.

5.4.2.1 Features Marked on Verbs

Kannada has complex verb formation rules.The various features of the Kannada verb is listed in table 5.5.

Table 5.5. Verb Features and Characteristic Suffixes

Features	Characteristic Suffixes	Example
Infinitive	Alu	maaDu+alu→maaDalu (to do)
Imperative	i,iri,oo,ee	maaDu+iri→maaDiri (you do)
Negative Imperative	beeDa,baaradu,kuuDadu	maaDu+alu+beeDa→maaDabeeDa (Don't do)
Finite forms Without GNP	Listed in Table 5.13	maaDu+ali→maaDali (let him do)
		maaDu+ooNa→maaDooNa (let us do)
Finite forms With GNP and Tenses	Listed in Table 5.13	
Verbal aspect Auxiliaries	haaku, biDu etc. (table 5.6)	maaDu+i+haaku→maaDihaaku (do non finite + put finite)
Reflexive	koLLu	
Causative Suffix	Isu	maaDu +isu →maaDisu (get it done)
Verb non Finite forms (Future Relative participle)	Table 5.12	maaDu+uv+a→maaDuva (we shall do it) maaDa+i→maaDi (after doing) maaDu+ade→maaDade (without doing) maaDu+id+are→maaDidare (did they do)
Conditional Suffix	are	maaDu+ade→maaDade
Imprecate	a	maaDu +a → maaDa

Inclusive clitic	uu	----
Emphatic clitic	ee	maaDu+utt+aane+ee→maaDuttaaneyee? (will he do?)
Interrogative clitic	aa	maaDu+uva+anu+aa→ maaDuvanaa (will he do?)
Indefinite clitic	oo	maaDu +id +anu+oo→ maaDinoo has he done ?)
Modal Auxiliaries	Table 5.7 shows the list	

5.4.2.2 Auxiliaries in Kannada

Auxiliaries in English are also called as **helping verbs** helping verbs are as *will, shall, may, might, can, could, must, ought to, should, would, used to, need* etc. These are used in conjunction with **main verbs** to express shades of time and mood. The combination of helping verbs with main verbs creates what are called verb phrases in English. In English *shall* is used to express simple future. Forms of has, 'how' are used to express tenses like present perfect and past perfect. There is also a separate section on the Modal auxiliary such as *can, could, may, might, must, ought to, shall, should, will*, and *would*, they do not change form for different subjects, Say, "I can write". Modal auxiliary expresses various meanings of necessity, advice, ability, expectation, permission, possibility, etc.

Kannada auxiliaries can be divided into aspect auxiliaries and modal auxiliaries. However, the auxiliaries in English occur as free morphemes and are easy for analysis. But in Kannada they occur as bound morphemes suffixed to the verb with which they occur. The general occurrence of aspect auxiliary is main verb (past verbal participle form) + Aspect auxiliary verb. **Another important distinction between Kannada auxiliary and English auxiliary is, the verbs which are acting as aspect markers are also used as main verbs in Kannada.** This is not true for English. Kannada has a set of verbs which may be added to verbal participle to give certain semantic nuances to the meaning of the sentence. Aspect markers are very similar to main verbs in their morphology and syntax. In fact they are derived from certain main verbs. But semantically they do not express the lexical meaning as their main verbs express in their aspect auxiliary usage. **The aspectual biDu 'completive' does not mean the same as the main verb biDu "leave"**. Consider an example of verb formation by adding a set of auxiliaries as shown in table 5.5. In Kannada thousands of such verb forms can be generated.

In Kannada the verbs which are acting as aspect auxiliary are actually main verbs. Apart from their usage as main verb, these verbs are acting as auxiliaries also. Consider an example.

Kannada: **ಅವನು ಕಥೆ ಬರೆದುಕೊಟ್ಟನು.**

Transliteration: **avanu kate baredukoTTanu.**

English: H**e wrote a story for someone's benefit.**

ಬರೆದುಕೊಟ್ಟನು (baredukoTTanu) word form is complex, derivative, formed by combining of nonfinite form first verb finite form of following second verb. Formation is shown below.

(Wrote)+(gave) + past of give + person 3 masculine = he wrote for somebody.

ಬರೆ+ ದು+ ಕೊಡು + ಭೂತ ಕಾಲ=ಬರೆದುಕೊಟ್ಟನು

bare+d+u+koDu+Past+p3.M.SL= baredukoTTanu

$\Leftrightarrow$

$\Downarrow$

Past Verbal Participial form + Aspect Marker

Here 'baredukoDu' looks as simple as single verb. But it is a complex verb, which is formed by the addition of aspect auxiliaries 'koDu' (give) to the 'baerdu' form (after writing) form, which is a nonfinite (past verbal participle form) of verb bare (write). **This kind of formation leads to lakhs of complex verbs.** But in reality there are only around 2,000 basic verb roots which are stored in the dictionary. Another thing is the complex verbs formed by this process follow the TAM (tense, aspect and modality) inflections of the auxiliary verb but not first verb. However, the meaning of such formation is inferred from the first verb itself. In the below example in table 5.5 the word form "tiMdubiDu" (eat up completely), the meaning is ate itself, it is not like 'ate and then left'. **This process of complex verb formation is productive and regular in Kannada.** It is not wise to store such complex verbs in the dictionary, instead we have handled the formation of such complex verbs through our morphology. In current existing morphological systems these verbs are stored as basic verbs and kept in the dictionary. **Another specialty of Kannada is more than one aspect marker can be attached, this is not the case with English.** Consider an example.

Example of Complex word formation in Kannada Input is given below:

ಮಾಡು+ಇ/ಉ(ಭೂತಕಾಲ)+ಕೊಡು+ಇ/ಉ(ಭೂತಕಾಲ)+ಹೋಗು+ಇ/ಉ+ಬಿಡು+ಇ/ಉ(ಭೂತಕಾಲ)+ಬಿಡು→ ಮಾಡಿಕೊಟ್ಟುಹೋಗಿಬಿಟ್ಟಬಿಡು. (2)

Transliteration:

maaDu (do)

+i/u (pastparticiple) (after doing)

+koDu (give)

+i/u (past participle) (after giving)

+hoogu (go)

+i/u (past participle) (be gone)

+biDu (leave)

English Equivalent: Leave after doing (this).

In example 2, 4 aspect markers are added. One can observe the complexity in word formation with respect to Kannada. The meaning of this phase can be inferred from the first verb maaDu 'do'. TAM inflections, PNG (person, gender, number) inflections follows last verb biDu.

Table 5.6. Handling of Aspect Auxiliaries

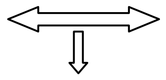
S.no	Aspect Marker	Aspect Meaning	Lexical Meaning
1	biDu	Completion	Leave
2	hoogu	Completion	Go
3	aaDu	Continuity	Play
4	koDu	Benefactive	Give
5	nooDu	attemptive	See
6	Haaku	Exhaustive	Put
7	koLlu	Reflexive	Purchase
8	Iru	Perfective	Be

5.4.2.3 Modal Auxiliaries

Modal auxiliaries contribute different shades of grammatical meaning. The different possible modal auxiliaries are shown in table 5.7. Modal auxiliaries are attached in infinitive 'al' form.

Do + to + May → May be done.

ಮಾಡು + ಅಲ್+ ಬಹುದು→ಮಾಡಲುಬಹುದು ... (3)



Infinitive form. + Modal Auxiliary→ Complex verb form.

Table 5.7. Handling of Modal Auxiliaries

S.no	Modal Auxiliary	Lexical Meaning
1	Beeku	Must /Want/need/ought
Consider an example Kannada :ನಾನು ನೋಡಬೇಕು (1) Transliteration: naanu nooDabeeku English Equivalent: I want to see		
2	kuuDadu(PROB)	Should not/no
Consider an example Kannada:ಅವನು ಹೋಗಕೂಡದು (2) Transliteration: avanu hoogakuuDadu English Equivalent: He should not go		
3	beeDa (NEG IMP)	Negative imperative/donot
Consider an example Kannada: ನೀನು ಬರಬೇಡ (3) Transliteration: niinu barabeeDa English Equivalent: you do not come		
4	Bahudu (PERM)	May/can/might

Consider an example Kannada:ಅವನು ಓದಬಹುದು (4) Transliteration: avanu odabahudu English Equivalent: he may read		
5	aara (NCAP)	Non capable/cannot
Consider an example Kannada:ಅವನು ಬರಲಾರನು (5) Transliteraion: avanu baralaaranu English Equivalent: He cannot come		
6	balla (CAP)	Capable/can
Consider an example Kannada:ಅವನು ಜಯಿಸಬಲ್ಲನು (6) Transliteration: avanu jayisaballanu English Equivalent : He is capable of winning		
7	paDu (PASS)	Paasive construction/Going to be
Consider an example Kannada:ಮನೆ ಕಟ್ಟಲಪಡುತ್ತದೆ (7) Transliteration: mane kaTTalpaDuttade English Equivalent: House is going to be built		
8	aagu (PASS)	Passive/going to be
Consider an example Kannada:ಮನೆ ಮಾರಲಾಗುತ್ತದೆ (8) Transliteration: mane maaralaaguttade English Equivalent: House will be sold		

The modal auxiliary in example 7 and 8, aagu (become) and paDu (happen) denote passive constructions. There are 3 tenses in Kannada, viz. Present, past and future tense and 3 persons (first person, second person and third person) and 2 numbers (singular / plural) and 3genders (Masculine, feminine and neuter). PNG inflections for verb are shown in table 5.8.

Table 5.8. Kannada PNG Markers

Person	**Number**	**Gender**	**Tenses In Kannada**			
			Present (utta)	**Future (uva)**	**Past (id)**	**Contingent**
First (P1_MFN_Singular)	Singular	Masculine /Feminine	eene	enu	enu	(y)eenu
(P1_MFN_Plural)	Plural	Masculine /Feminie	eeve	evu	evu	(y)eevu
Second (P2_MFN_Singular)	Singular	Masculine /Feminie	ii(ye)	e	e	iiye
(P2_MFN_Plural)	Plural	Masculine /Feminie	iiri	iiri	iiri	iiri
Third (P3_M_Singular)	Singular	Masculine	aane	anu	anu	(y)aanu
(P3_F_Singular)		Feminie	aale	alu	alu	(y)aalu
(P3_MF_Plural)	Plural	Masculine /Feminie	aare	aru	aru	(y)aaru
(P3_N_Singular)	Singular	Neuter	ade	adu	itu	iitu
(P3_N_Plural)	Plural	Neuter	ave	avu	avu	(y)aavu

The contingent future is special to Kannada. This form does not exist in other south Indian languages like Tamil, Telugu. Verb forms like maaDiyaanu 🡐maaDu+i+y+*aanu* (he might do) refers to the contingent future. This feature is not handled in (Baskarann, 2008). This form is formed by adding the suffix *yaanu* to the past verbal participle form. Morphological suffixes for contingent form are shown in table 5.9.

Table 5.9. Handling Morphology of Contingent finite Forms

Gender/Number/Person	**Suffix**
Contingent _P1_MFN_Singular	yeenu
Contigent P1_MFN_Plural	yeevu
Contingent P2_MFN_Singular	iye
Contingent P2_MFN_Plural	iiri
ContingentP3_MF_Singular	yaanu/yaalu
Contingent P3_MFN_Plural	yaaru
Contingent P3_N_Singular	iitu
Contingent P3_N_Plural	yaavu

Aspect denotes the status of the activity, durative aspect indicates the progressiveness, and perfective aspect indicates completion. Aspect marker suffixes are shown in table 5.10.

Table 5.10. Morphology of Aspect System

Aspect	**Marker**
Simple	-
Durative	Uttiru
Iterative	Uttiruttiru
Perfective	Iru

Set of particles can be added to word categories irrespective of their category. Unlike noun are restricted by case inflections. The following table 5.11 gives the list of such particles.

Table 5.11. Particles Applicable to all Categories

Description	**Suffix**
particle varege (Till)	ವರೆಗೆ, varege
particle atta (that side)	ಅತ್ತ,atta
particle aMte (similarative)	ಅಂತೆ, aMte
particle tanaka (Till)	ತನಕ, tanaka
particle maTTige (As concern)	ಮಟ್ಟಿಗೆ, maTTige

particle ashTu (that much)	ಅಷ್ಟು, ashTu
particle ashTuu (including that much)	ಅಷ್ಟೂ, ashTuu
particle ashTe (as much as)	ಅಷ್ಟೆ, ashTe
particle ashTee (as much as in emphatic)	ಅಷ್ಟೇ, ashTee
particle eenu (what)	ಏನು, eenu

Verb non finite forms are those that cannot stand as main verb of the sentence and rarely occur before the full stop. Morphology suffixes of such non finite forms are shown in table 5.12. The explanation of these non finite forms is given in chapter 3. Under non finite verbs.

Table 5.12. Morphology of Verb Non Finite Forms

Type	**Suffix For Rules**
Conjunctive Participle	ಇ/ಉ (i/u)
Relative Participle	ಅ (a)
Infinitive	ಅಲು (alu
Negative past conditional participle	ಅದೆ(ade)
Conditional	ಎ (e)
Conditional2	ಅರೆ(are)
Conditional3	ಒಡೆ (oDe)
Concessive	ಅರೂ (aruu)

In Kannada there is a set of finite verbs which have gender, number and person inflections. These behaves the same for all gender/number/person suffixes. The table 5.13 shows the list of such verbs. Detail explanations of these forms have already been given in chapter 3, under section of finite verbs.

Table 5.13. Verb Finite Forms without GNP Suffix

Description	Suffix for Rules
Optative	ಅಲಿ (ali)
Optative_Conjunctive	ಅಲೀ(alii)
Optataive_Interoogative	ಅಲಾ (alaa)
Optative _Interoogative	ಅಲೇ(alee)
Hortative Plural	ಇರಿ (iri)
Plural imperative	ಓಣಾ (ooNa)
Plural imperative vocative feminine	ಇರೆ (ire)
Plural imperative vocative masculine	ಇರೋ (iroo)
Singular Imperative	ಉ(u)

Vocatives: This category of suffixes are applicable to all categories and can be added to give emphasis, for example naaneenappa. The following is the list of vocative suffixes.

- appa(a) vocative
- amma(a) vocative
- ayya , phi null

5.5 Handling of Derivation Morphology

Derivation is the process of formation of new words or inflectable stem from another word or stem. It typically occurs by the addition of an derivation suffix. The derived word belongs to different word class from the original. It may thus take the inflectional affixes of the new word class.

- **Derivation has the following characteristics.**

•Typically produces a greater change of meaning from the original form.

•Is more likely to result in a form which has a somewhat idiosyncratic meaning.

•Often changes the grammatical category of a root.

•Kindness is derived from kind.

Here are some kinds of derivational operations in English:

•Operations that change the grammatical category of a root. Verbs and adjectives are derived to nouns: Consider examples like amaze→ amazement, speak →speaker, perform→ performance, soft→ softness, warm → warmth.

- **Kannada Derivational Morphology**

Derivation morphology deals with change of category from one to another as shown in below figure. A noun is derived from a verb or adjective, we have shown a sample of addition of suffix to the verb root.

5.5.1 Nominalizers: nominalizers are used to derive nouns from verbs. Consider an example.

➢ Use of derivational suffix ವಿಕೆ "vike" to derive nouns from verb.

maaDu 'do' (verb) +vike→'maaDuvike' (noun) "process of doing".

➢ Causation Kannada suffix ಇಸು "isu" to derive verb from noun.

prasne'question'+ isu →'prasnisu' is a verb which means questioning.

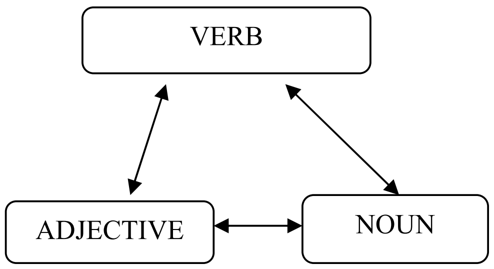


Figure 5.4. Derivation Morphology

5.5.2 Verbalizers: A set of verbs which are used to derive verbs from nouns and adjectives. There is a set of verbs like 'aagu', biDu iru, goLLu, aaDu, hoDe, maaDu which are used in deriving verbs from nouns.

Noun+verb→Verb. Consider the following examples gaaDi+hoDe→ gaaDihoDe

nidde+maaDu→niddemaaDu

- **Use of verb ಆಗು (aagu) 'become' in verbalization.**

Most of the nouns in Kannada are derived as verbs by the addition of the verb 'aagu' to basic noun roots. The following examples illustrate the derivation Process.

Noun to verb derivation using ಆಗು (aagu) verblizer.

a). peTTu (Injury) 'noun' + aagu 'become/happen'→peTTaagu
maLe (rain) 'noun' +aagu→ maLeyaagu (raining)
beLe (crop)'noun' + aagu→beLeyaagu (grow up as a crop)
paasu (pass) 'noun' +aagu→paasaagu (pass)
ishTa (like) 'noun' + aagu→ishTavaagu (be liked /be to one's liking)

- **Adjective to Verb Derivation Process**

b). haaLu 'adjective'+ aagu →haaLaagu (get ruined)
suMdara 'beautiful' + aagu→suMdaravaagu (become beautiful)

- **Use of Verb aagu in Complex verb Derivation**

In Kannada we have a set of verbs which do not have aspect and tense inflections. Like 'beeku' (want), beeDa (donot to want), kuuDadu (no), bahudu etc. By adding verb "go" to these verbs aspect and tense inflections can be added.

- **Usage of aagu with Modal auxiliaries**

The modal auxiliary beeku (want) is a verb, 'beeku' + aagu→beekaagu (wanted). Now 'beekaagu' is a compound verb. We had another set of conflicts like whether maataaDu 'talking' (maatu + aaDu is a noun+verb) a is compound, whether such words be stored in the dictionary, or generated by morph system. Verbalizers like aagu iru, aaDu, paDu, goLLu plays an important role in generation of many compound verbs. Kannada language has a complex morphology, many suffixes get added to form the complex inflected root. Consider the formation of verbal noun in the example shown below.

ಮಾಡು+ಅಲ್ +ಬೇಕು+ ಆದ+ ಅವರು= ಮಾಡಲುಬೇಕಾದವರು

maaDu + al + beeku + aada + avaru = maaDalubeekaadavaru

do + to +must + happen +they= One who is needed for doing

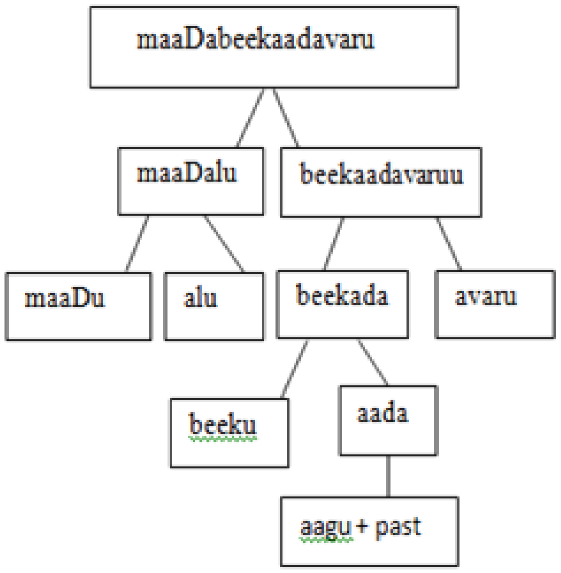


Figure 5.5 Sample Derivation Tree for Verbal Noun

5.5.3 Compound Morphology

Compounding is a word formation process, whereby two or more words are combined to produce a new word. The interesting property is meaning of the resulting word cannot be understood as the combination of meaning from the combined words. Compounds yield new meaning rather than its constituents.

External saMdhi: In Kannada there is set of polar verbs which act as verbalizes and produce new set of verbs ‘maataaDu’ (maatu+aaDu) 'talking' (noun+ verb) compound. Whether the verbs generated by this process should be stored in the dictionary or not was a question later we decided morph system to handle these. Other verbs like aagu iru, aaDu, paDu, goLLu etc., act as verblizers.

Compounding involves two or more words rather than affixes. Syntactic rules are not considered in the internal structure of compound words. In syntax morphologically complex words are not different from simple words. Internal structure of compounds is inaccessible to syntax. A compound word is formed by a combination of two words and acts as independent entity and allows no inflections in between the two elements. These words do-not allow inflection between the conjugating words. Hence compounds are syntactic atoms like other words. Processing the meaning from compounds from its constituents is not always easy.

```
Noun+ noun compounding :
kudare +gaaDi = kudaregaaDi                        (1)
(horse) +(cart) =  Horse cart

Noun +  verb compounding:
juuju + aaDu   =  juujaaDu                         (2)
(gamble)+(play)=(gambling)
kai    +  biDu = kaibiDu                           (3)

adjective+ verb compounding:
kappu + aagu =   kappaagu                          (4)
(black)+(become)= (become sad)

Verb     + verb compounding:
oDu  + aaDu  =   oDaaDu                            (5)
(run) + (play) =   (run around)

adjective +noun compounding:
kappu + halage =    kappuhalage                    (6)
black)+ (board) = (black board)
```

5.6 Morphological Generation

Morphological generation is the inverse of morphological analysis. In analysis, the goal is to find the mapping of *(inflection)*→ *(root, POS)*. Where as in generation, one finds the reverse mapping, *(root, POS)* $\rightarrow$ *(inflection)*. This relation is actually symmetric as *(root, POS)* $\rightarrow$ *(inflection)*. Because of this bidirectionality, the root-inflection pairs discovered in morphological analysis can correctly be used directly as the root inflection pairs for morphological generation. Morphological analysis is a many-to-one mapping from inflection to root, generation is a one-to-many mapping. A natural extension to this work involves extracting the syntactic features associated with each inflection. This need be done either in conjunction with a part of speech tagger, or a parser, or syntactic feature extraction. While the models are reasonably effective at analyzing the agglutinative inflected forms. For a task such as machine translation, fine-grained part-of-speech tags such as verb, 1st person, and singular, present, indicative are necessary. For many other tasks, such as information retrieval or word sense disambiguation, coarse-grained POS tags such as v-verb or n-noun are sufficient. ***We prove here our hypothesis that hierarchical tagging has wide application than flat tagset.*** The basic aim of this analyzer/generator is to initiate some resource building and perform language processing. The maximum inflections *we can have for a verb 'do' in English is about 5 different forms like do, does, did, done, doing. But when we compare the possible inflections forms for the same verb 'do' in Kannada is more than 3000 plus different forms are observed. One can see the morphological inflection complexity of Kannada language. Similarly the noun in English has a maximum of 2 inflections for word say boy, boys. But the same noun in Kannada has around 250 different inflections.*

5.7. Comparisons of Proposed Morphological Analyzer generator (MAG) system with Existing MAG systems

Few existing systems have following draw backs as discussed below.

(Shambhavi, 2011), proposed "**Kannada Morphological analyzer and generator using trie".** There are many aspects left unhandled in Shambhavi work. The Limitations of Shambhavi work in comparisons with our work are listed below.

(Shalini et al., 2007) proposed "Development of prototype for morphological analyzer for south Indian language of Kannada". This is only prototype. This work also has following limitations.

Limitations of the above addressed similar works.

- Lack of handling of auxiliary verbs which play important roles in formations of laths of verbs. Since the real verbs stems are only 1000. But there are lakhs of verb forms possible only by handling auxiliaries. This is very important factor.
- Clitics morphology is not handled in these morphological analyzers
- Only flat tag set is used. Hence this morphogical analyzer can not capture detailed information.
- Lack of handling of modal auxiliaries. This factor reduces verb tagging rate compared to our method.
- Derivational morphology is not touched. Only simple inflectional morphology is implemented.
- Paradigm approach is used.
- Only 8 forms of simple inflection verbs are handled where as there more than such 3000 verb forms which are left unhandled example like hoogibiDu (go completely) aspect auxiliary formations. Hoogabeeku (needs to go) such forms are not handled. Size of dictionary is just 3700 words in Shambhavi work.
- Handling of gerunds are not discussed.
- Causative aspect verb formation not handled. This factor reduces verb tagging rate compared to our method.
- Countable and uncountable feature information are useful in generation otherwise leads to over generation is not highlighted.
- Use of verbalizers in coining new verbs is an important aspect not handled in above works listed so far.

Our morphology system is the first system built using hierarchical tag set among Indian Languages. Size of dictionary is more than these work in our proposed AMAS work i.e. around 30,000. above mentioned limitations are have overcome in our work.

5.8 Experiments and Results

The Morphological analyzer is experimented with sample files from *DoE CIIL* corpus which is in *ISCII* format. We have developed a new roman transliteration scheme called "*lerc-uoh",* for the conversion of ISCII to Roman format. Morphological analyzer reads the inflected surface form of each word in a text and writes its lexical form consisting of canonical form of the word and a set of tags showing its syntactic category and morphological characteristics. This module takes sentence as input, tokenizes them into words and looks up each word in the dictionary for its corresponding POS category/categories. If not identified then it is sent to morphological analyzer to check for inflections and to assign the possible tag. If morphological analyzer does not tag, then it will be probably proper noun and is passed to NER module (named entity recognizer).

1. **We have used *DoE CILL* corpus of 3 million words for experimentation**, a sample file *Bank1.aci.out* is considered, the size of the file is *1084* words. We observe the following derivation types in the corpus as shown in box 5.6. More than *50 % words in the corpus are derivational* types. This shows the importance of handling the derivational morphology. The performance is expressed in terms of precision, recall and F-measure. Shown in table 5.14.

Derivation Types		No. of words
n > v	…	148
n > adj	…	115
Gerund	…	115
Compound	…	150
v > adj	…	80
v > adv	…	145

Box 5.6: Derivation Types in Bank1.aci.out

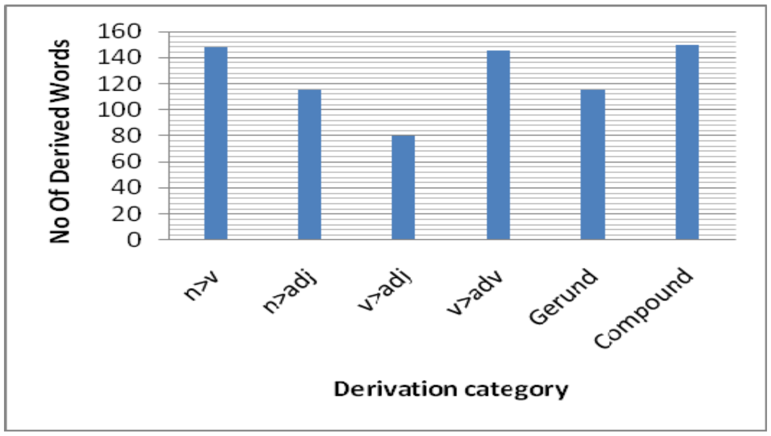


Figure 5.6. Distribution of Derived Words in Bank1.aci.out File

Recall= $\frac{TP}{(TP+FN)}$ …(5.1)

Precision= $\frac{TP}{(TP+FP)}$ …(5.2)

F-Measure = $\frac{2*Precision*Recall}{Precision+Recall}$ …(5.3)

Table 5.14. Showing Recognition of Morphological system Efficiency

Total Derivation Words	**842**
Correct Recognized	700
Wrong Recognized	53
Not Recognized	89
Recognized	753
Precision	95%
Recall	92%
F-measure	93%

We have attempted towards developing a derivational morphological analyzer for Kannada, this tool is important in morphological process. Results regarding derivational morphology are good.

2. Similarly to check the performance of morphological generator, *we manually generated gold standard data for nouns by choosing human, non- human combinations from a/e/i/o/u/ word endings and it is observed that for each stem ending around 250 inflections can be generated, so 6*250 combinations are possible these* cover all inflections identified, these are crossed checked with manually generated and all forms are matched with inflected forms generated by morphological generator. We observed that more than 95 % words are recognized correctly. Similarly we prepared gold standard data of 3000 verbs for verbs and observed that the recognition rate is around 85 %. The inflections generated by morphological generator are shown in figure 5.7.

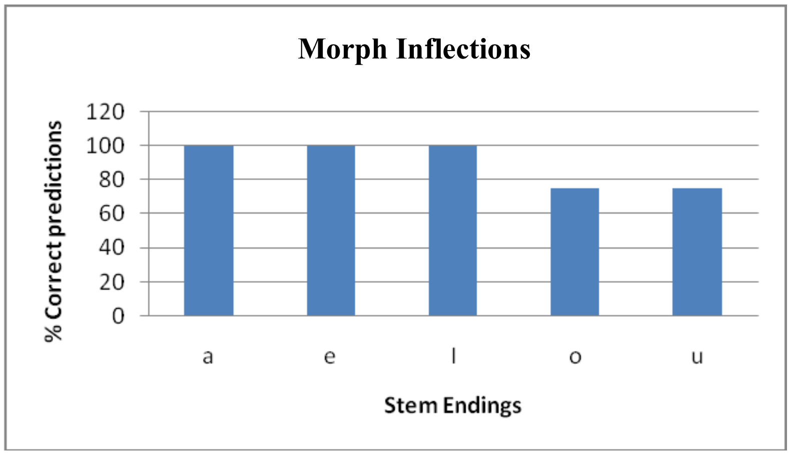


Figure 5.7. Recognized Morph Inflections Based on Stem Endings

Total countable nouns in our hierarchical dictionary are 23621, so the possible inflections for these words is 23621 250=59,05,250 (fifty nine lakhs, five thousand, two hundred fifty words) inflections, similarly the total number of uncountable nouns is 707 and the possible inflections are 707 *125=88375 (88K), uncountable* nouns have no plural inflections. Morphological generator handles the inflections correctly. But is over general at instances like past conjunctive participle branch. To avoid the over general cases we have limited the addition of auxiliaries to only 6 levels.

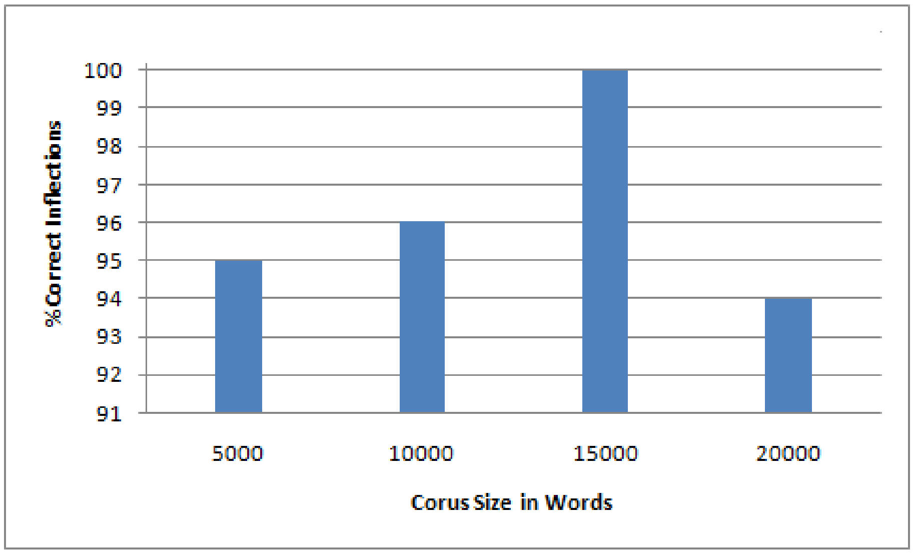


Figure 5.8. Corpus Size versus Morph generator Performance

We developed morphological generator first later developed analyzer. In order to check the whether the all the possible inflections generated by morph generator are correct are not we divided the dictionary file in terms of word-size as top 5000 words, next 10,000 words and selected 15,000 words and finally 20,000 words from dictionary and generated all possible inflections and observed that correct predictions were less when size of file was 20,000 words. This was due to non marking of "real u" information in the dictionary. Results with respect to morphological generator are encouraging. Though morphological generator is not used for annotation process, it is a useful component in Machine Translation systems.

3. Kannada Sample file Account4.aci.out file from DoE CILL Corpus is selected for analysis and it is observed that many words in the corpus are formed with auxiliaries and also modal auxiliaries as depicted in figure 5.9. The analysis varies with the type of corpus.

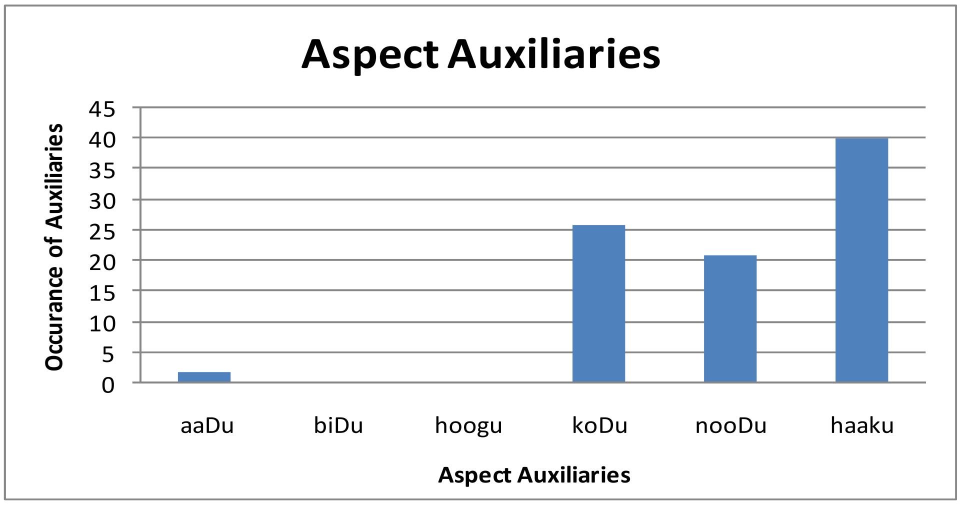


Figure. 5.9. Occurrences of Aspect Auxiliaries in File Account4.aci .out

In the above corpus the auxiliary 'haaku' 'put' has occurred maximum number of times followed by nooDu and koDu, auxiliaries biDu and hoogu have not occurred at all. The distribution of modal auxiliaries in Account4.aci is shown in figure 5.10.

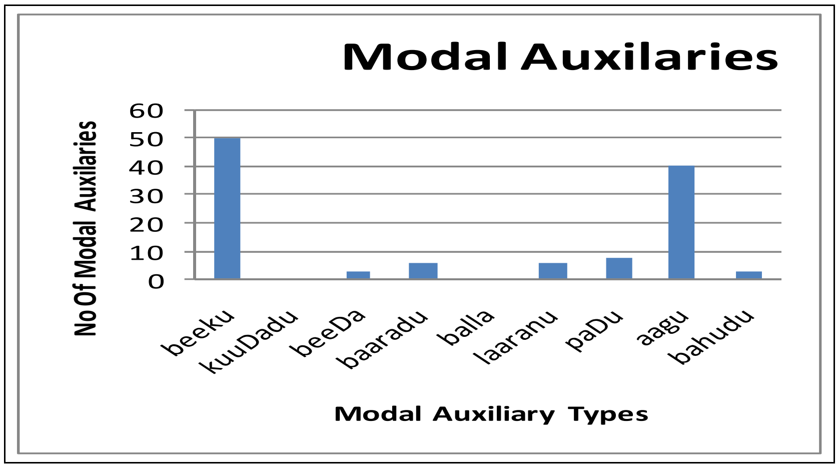


Figure. 5.10. Occurrence of Modal Auxiliaries in File Account4.aci .out

Figure 5.10. Shows that auxiliary ‘beeku’ (must) have occurred maximum number of times followed by aagu, paDu, and kuuDadu has not occurred. This variation depends on the corpus.

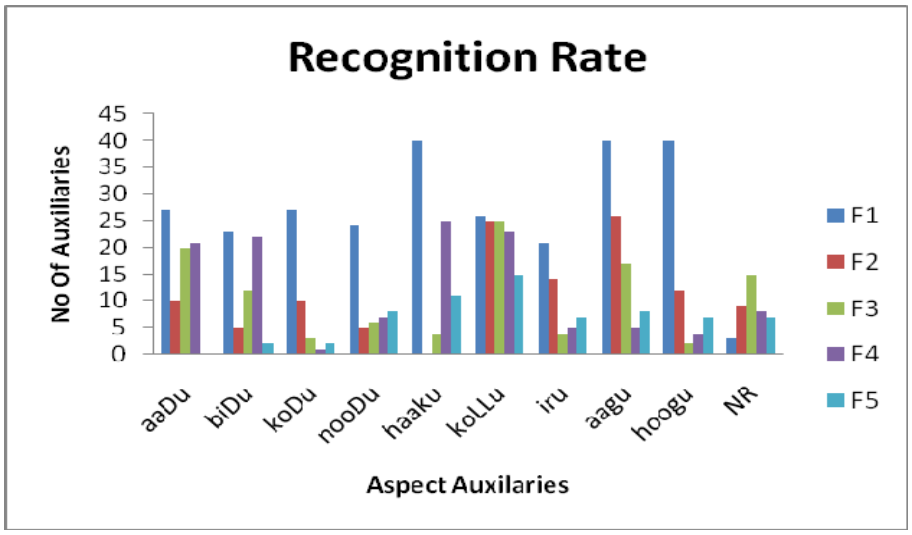


Figure 5.11. Aspect Auxiliaries Distribution in CIIL Corpus 5 Sample Files

Figure 5.11 shows the distribution of aspect auxiliaries in 5 sample files F1, F2, F3, F4, and F5. The files are of varying size. The morphology of Kannada becomes complex due to occurrence of auxiliaries as bound morphemes with the root or derivative stem, whereas they occur as free morphemes in English. We observe that 5 to 6 levels of aspect marker can be added to the derivate stem to form a complex verb root in Kannada. Auxiliary verbs help the main verb to denote the actions of the subject. They help in making compound words and passive voice statements. Auxiliaries are useful in serial verb construction. Morphology of auxiliary verbs like aspect auxiliaries and modal auxiliaries play a very important role as far as a Dravidian language like Kannada is considered. Our Morph module handles the auxiliaries.

1. Similarly the morph module was tested on the first 5,000 most frequent Kannada words of Department of electronics (DoE), Central Institute of Indian Languages (CIIL) corpus. Words selected included all the nominal, pronominal, adjectival and verbal inflections of the Kannada language. Around 80 % words were analyzed correctly, manually checked and verified; remaining 20% words were a mixture of spelling variations, dialect variations, and compound words. Hence such words in the raw corpus selected for testing are not analyzed. A few words are not analyzed due to missing entries in the dictionary.

2. We extracted 3119 different word forms for the verb maaDu "do" from Kannada Sahithya Parishat corpus and ran our morph over it and observed the following details. We observed 9 cases of ambiguities for the verb maaDu.

```
3119 words seen. Overall Performance: (87%)
1 word(s) found in the dictionary: (0 %),
Verb 'maaDu' found in the dictionary
2717 word(s) analyzed by Morph: (87 %)
401 word(s) could not be analyzed: (13 %)
624 word(s) had more than one analyses. (20 %)
Average Ambiguity per Word: 1.30
Maximum number of analyses: 9
```

- Ambiguity not Resolved By Our Morph

Honorificity cannot be resolved at morph level but can be handled at syntax level. We observed that the following ambiguity cases which cannot be resolved at morph level. In the below cases the pos tagger should handle these ambiguities; disambiguation rules are to be designed based on context. We have proposed the following disambiguation rule by looking at the person agreement and subject. When we ran the different forms of the verb maaDu ’do’, we observed the following ambiguity cases as shown in box 5.8. All 250 noun forms are analyzed correctly for the word huDuga (boy). A sample output ofmorphological analyzer for noun "boy". huDuga is shown below in box 5.7.

```
1: huDuganu 1: [huDuga: N-COM-COU-MS-NOM] : singular -
nominative -null -
2: huDuganuu 1: [huDuga: N-COM-COU-MS-NOM] : singular -
nominative -inclusive - null
3: huDuganee 1: [huDuga: N-COM-COU-MS-NOM] : singular -
nominative -emphatic - null
4: huDuganoo 1: [huDuga: N-COM-COU-MS-NOM] : singular -
nominative -indefinite - null
5: huDuganaa 1: [huDuga: N-COM-COU-MS-NOM] : singular -
nominative -interrogative -
6: huDuganuunaa 1: [huDuga: N-COM-COU-MS-NOM] : singular -
nominative- inclusive - interrogative
7: huDuganeenaa 1: [huDuga: N-COM-COU-MS-NOM] : singular -
```

Box 5.7 Sample Morphological Analyzer Output for Noun Boy

A Sample morph output for the verb ’Do’ maaDu and its ambiguity cases is shown in box below.

```
1: maaDide
     1: [maaDu: V-] : 15 +i_u= cjp 30
     +ide=perfective_present_p3_n_singular 91 98 100
     2: [maaDu: V-] : 15 17+id= past 20 +e= p2_mfn_singular
     91 98 100
     3: [maaDu: V-] : 15 17+id= past 20 +e(nu)=
     p1_mfn_singular 91 98 100
2: maaDisi
     1: [maaDu: V-] : +isu= causative 15 17
     +i(ri)=plural_imperative 91 98 100
     2: [maaDu: V-] : +isu= causative 15+i_u= cjp 91 98 100
3: maaDaballa
     1:[maaDu: V-] : 15 17 +al(u)= infinitive 25 26 +balla=
     rp_capabilitative100
      2: [maaDu: V-] : 15 17 +al(u)= infinitive 25 26
     +ball(a)=capabilitative 100
4: maaDabayasuve
      1: [maaDu: V-] : 15 17 +al(u)= infinitive 25 26
     +bayasu=asp_aux_bayasu 15 17 +uv= future 22 +e=
     p2_mfn_singular 91 98 100
     2: [maaDu: V-] : 15 17 +al(u)= infinitive 25 26
     +bayasu=asp_aux_bayasu 15 17 +uv= future 22 +e(nu)=
     p1_mfn_singular 91 98100
5: maaDadee
      1: [maaDu: V-] : 15 +adu=
     future_negative_p3_n_singular 91 +ee=emphatic_clitic 95
     98 100
     2: [maaDu: V-] : 15 17 +a= negative 35+de= cjp 92 +ee=
     emphatic_clitic 95 98 100
```

Box 5.8 Ambiguity Analysis Given By Morphological Analyzer

```
6: maaDadaMte
      1: [maaDu: V-] : 15 17
      +adaMte=negative_similaritive_aMte 91 98 100
      2: [maaDu: V-] : 15
      +adu=future_negative_p3_n_singular 91 +aMte=
      hear_say_clitic 98 100
7: maaDalaaguvaMte
      1:[maaDu: V-] : 15 17 +al(u)= infinitive 25 26
      +aagu= asp_aux_aagu 15 17+uv= future 22 +aMte=
      similaritive_aMte 91 98 100
       2: [maaDu: V-] :15 17 +al(u)= infinitive 25 26
      +aagu= asp_aux_aagu 15 17 +uv= future 22 +aMte=
      sub_conj_aMte 100
8: maaDe
      1: [maaDu: V-] : 15 17 +e= conditional100
      2: [maaDu: V-] : 15 +e=
      future_negative_p2_mfn_singular 91 98100
9: maaDalee (i):
      1: [maaDu:N-COM-COU-NS-NOM-NULL||V-] : 15 17
      +alee= optative_interrogative100
      2: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : 15 17
      +alu= infinitive95 96 +ee= clitic_ee 97 98 100
```

Box 5.8 Continuation of Ambiguity Analysis by Morphological Analyzer

1. Similarly *14000* different inflected word forms of pronoun are generated manually and ran through morph and all the word forms are analyzed correctly. We got 100% correct analyses.

We conducted the experiments to evaluate the system for nouns separately. The system was evaluated based on the parameters of precision and recall, Precision. The observed results are listed in table 5.15. We have also experimented 10,000 nouns from kannada sahithya parishat corpus called as kvk corpus and ran morph system on this corpus and observed the following results.

Table 5.15. Output for Nouns of KVK Corpus

Nouns (10,000 input words) Results	
True positives	9600
True Negatives	180
False Positive	200
False Negatives	20
Precision	98%
Recall	97%
F-measure	97%

- ## Summary

We have developed an analyzer and generator for Kannada verbs, which analyses and give the grammatical category of the given word. Our morphological analyzer is the first system developed with hierarchical tagset approach. The system uses the hierarchical tagset, and dictionary developed using hierarchical tagset, use of the resources which gives fine grained information is a new attempt as far as Indian language is concerned. The system is developed using PERL on the LINUX. This system requires an extensive linguistic expertise. It is a system of representing detailed morph syntactic and semantic information in such a way that it is computationally useful. The morph module is main component of our automatic morphosyntactic annotator system (AMAS) architecture. Handling of spelling variations, few dialectic variations and to improve dictionary as per the need of hierarchical design of tag set are the extensions which can be performed later. We have covered all the aspect regarding clitic handling, derivational morphology, handling of aspect auxiliaries, modal auxiliaries which are widely distributed in the corpus. This is an exhaustive system existed so far for Kannada. Building morphological analyzer for the languages like Kannada having complex morphology is not an easy task.

Chapter 6

Named Entity Recognition and Classification Using Rule Based Method

In the previous chapter, proposed AMAS architecture is discussed. In order to handle proper nouns in the corpus there is a need Named Entity Recognition module. NER is developed as an additional component of AMAS architecture to enhance performance by tagging propernouns. NER is an important task in Natural Language Processing (NLP) applications like Information Extraction, Question Answering etc. In this chapter, we are proposing rule based methodology to recognize Kannada named entities. The named entities like name of person, location namec organization names, number, measurement and time are considered for recognition in this methodology. We have developed suffix and prefix list, set of regular expressions and proper noun dictionary of 5,000 words in rule based technique. Our results of recognition are encouraging and the methodology has the precision around 86%. The corpus of the widely circulated Kannada news paper called PrajaavaaNi is used for experimentation.

Capitalization feature is used in identifying named entities in English. But capitalization feature does not exist in Kannada and lack of capitalization makes NER task more challenging in Kannada as compared to English. Proper nouns are indistinguishable from different forms of common nouns and adjectives in Kannada. These features of Kannada also make NER a challenging task.

6.1 Introduction to NER

Named Entity Recognition is a task of finding and classifying proper names, location names, organization names, etc. in the given text. NER is considered an important task in many NLP applications, like machine translation, question-answering systems, indexing for information, information retrieval, data classification and automatic summarization etc. Information extraction (IE) is a vital activity in NER. Because of the importance of Information extraction (IE), DARPA (Defence

Advanced Research Project Agency) initiated a number of Message Understanding Conferences (MUC) in the mid nineties. According to the specification defined by MUC, the NER tasks generally work on six types of named entities like person name, location name, organization name, time, measurement and number. For example consider a sentence: **Mr. Rahul joined as manager of the Municipal Corporation in Bangalore on Monday 14, September 2004.** The various named entities in this sentence are "Mr. Rahul" is a person named entity", "Municipal Corporation" is an organization entity, "Bangalore" is a location entity" and "Monday 14, September 2004 is a time entity". In a given text it is necessary to recognize automatically the named entities.

Machine learning methods include conditional random field, support vector machine (SVM), Maximum Entropy Model, Decision Tree, Hidden Markov Models (HMM) etc. All these approaches make use of gazetteer information to build the systems, because gazetteer information improves the accuracy. In addition to machine learning techniques, one can use a rule based approach, which needs effort in devising rules in consultation with experienced linguists. In machine learning techniques, use of collection of annotated documents to train classifier is required. However, handcrafted rule-based systems generally give good results. But the challenge lies in development of rule- based techniques that require huge experience and grammatical knowledge of the language of interest.

From the literature survey, it is observed that not much work has been carried out on NER for Indian languages using a rule based approach. Proposed rule based approach is the first attempt as far as Kannada is considered.

6.2 Features Used for NER

Feature selection is important in named entity recognition. We have adopted different trial combinations to pick up suitable features taking into considerations all possibilities. We found that suffix and prefix information works well for an inflected language like Kannada. We have used a sliding window size feature, F= {w_{i-1}, w_i, w_{i+1}, w_{i+2} etc.}. We have manually prepared various gazetteer lists for use in NER, namely, 11 location suffixes, 49 designation prefixes, 74 organization prefixes, 50 person prefixes, 32 measurement prefixes, 32 next word clues, and 5,000 words proper-noun dictionary. The following features have most often been used for the recognition and classification of named entities.

i) **Context word feature**: Word previous and next to a given word are used as a feature. Rule 10 is used to catch this feature. Rules R1-R9 are also applicable for this context word features.

ii) **Dictionaries**: We have used proper noun dictionary for our NER task, wherein POS tags are attached to the words. Rule 16 is highlight this feature.

iii) **Named Entity Information**: Named Entity information is the feature in which named entity tag of the previous word is considered. It is the dynamic feature. Rule 13 is used for organization identification.

iv) **Gazetteer Lists**: Due to the scarcity of resources in electronic format for Kannada language, the gazetteer lists are prepared manually. Seven different lists are prepared.

v) **Word suffix and prefix**: Word suffix information is useful to identify NEs. Variable length suffixes of a word can be matched with predefined lists of useful suffixes for different classes of NEs. The different suffixes for person names are *gauDa, appa, swami,* etc., and for location names *baad, pura, haLLi* etc. Nested NEs *like akkamaadeevi raste*, is tagged as location NE, instead of proper-noun followed by a common noun. In this feature, a length of 1 to 4 words of the current word or the surrounding words is taken into consideration. Rule 11, 12 are used for this feature.

vi) **Digit information**: This feature is useful in date related kind of information, expressions, numbers, time, all ways of specification of time like 8.00 AM / am, named entity measurement like 10 kg, date information like 12 / 09 / 2009, 12-09-12,13-04-2012, 14 June 2009, Monday 14 January, representing floating point values like 12.345. The features like digits and percentage, digits and hyphen, digits and period, digits and slash are handled. Rules R1-R9 represent these features.

vii) **Word clue:** This feature helps in identifying next or previous token as NER. For example, raamaneMba huDuga. Its equivalent in English is ‘boy called raama'. Words derived as relative participles by adding suffix 'aada' for the animate words represent human beings like ‘avara magaLaada menaka'. The equivalent in English is ‘their daughter named menaka'. The word following the participle so formed is surely a person representing named entity.

viii) **Other cues:** The news paper corpus has token pattern like place name followed by date like Delhli.9 from these pattern one can identify location named entity. Rule 15 is used; rule 15 represents a regular expression like Delhi.9.

6.3. Proposed Rule Based NER Methodology

The architecture of a proposed automatic NER system is as shown in Figure 6.1. The methodology takes a Kannada sentence as input, recognizes the named entities and categories them. The methodology comprises four tasks, namely, formation of transliterated corpus, tokenization of sentences, dictionary lookup, and NER Module.

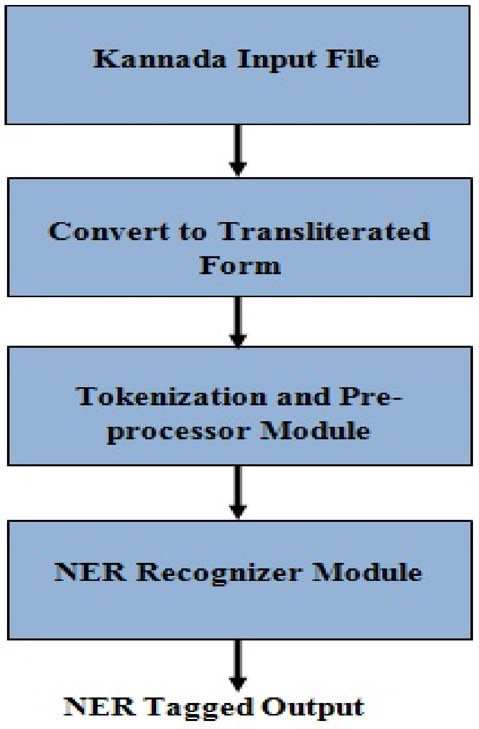

Kannada Input File
Convert to Transliterated Form
Tokenization and Pre-processor Module
NER Recognizer Module
NER Tagged Output

Figure 6.1. Named Entity recognition System Using Rule Based Method

6.3.1 Transliterated Corpus

The raw corpus is converted into transliterated corpus by a converter program ur.pl (Unicode to Roman Converter), which transliterates the Kannada words to Roman form. Thus, a transliterated form of the text given in box 6.2. Sample input is given in box 6.1.

ಇವತ್ತು ನವರಾತ್ರಿಯ ಮೂರನೆಯ ದಿನ.
ಮೈಸೂರು ಮತ್ತು ತಂಜಾವೂರು ಎರಡೂ ೧೮-೧೯ ನೆ ಶತಮಾನಗಳಲ್ಲಿ ದಕ್ಷಿಣ ಭಾರತದ
ಪ್ರಮುಖ ಸಾಂಸ್ಕೃತಿಕ ನೆಲೆಗಳಾಗಿ ರೂಪುಗೊಂಡವು. ಹಾಗಾಗಿಯೇ ಇಂದಿಗೂ ನಾವು ವೀಣೆ,
ಚಿತ್ರಕಲೆ ಮತ್ತು ಭರತನಾಟ್ಯ ಇವೆರಡರಲ್ಲೂ, ಮೈಸೂರು ಶೈಲಿ ಮತ್ತು ತಂಜಾವೂರು ಶೈಲಿ
ಎಂದು ಎರಡು ಪ್ರಮುಖ ಶೈಲಿಗಳನ್ನು ನೋಡಬಹುದು. ಕರ್ನಾಟಕ ಸಂಗೀತ ತ್ರಿಮೂರ್ತಿಗಳೆಂದೇ
ಹೆಸರಾದ ತ್ಯಾಗರಾಜ, ಮುತ್ತುಸ್ವಾಮಿ ದೀಕ್ಷಿತ ಮತ್ತು ಶಾಮಾಶಾಸ್ತ್ರಿ ಅವರು ಬಾಳಿದ್ದು
ತಂಜಾವೂರು ರಾಜ್ಯದಲ್ಲೇ. ಅದರಲ್ಲಿಯೂ ಶಾಮಾಶಾಸ್ತ್ರಿ ಅವರು ಇದ್ದದ್ದಂತೂ ತಂಜಾವೂರು ನಗರದಲ್ಲೇ.

Box 6.1: Sample Input Kannada Text

The Unicode text in Kannada is converted to Roman text, using transliteration program. The transliterated version is shown in box 6.2. The English translation is given in box 6.3.

Box 6.2: Transliterated Text for Kannada given in box 6.1

Today is navaratri's third day. Mysore and TaNjavora were established as two of both 18-19 centuries south Indian cultural centres. Even today, in playing viiNaa, art of painting and bharatanatya both Mysore and TaNjavora style may can be seen. OF the three people famous for Karnataka music, Tyagaraj, Muttuswami Diikshita and Syamasyastri lived in TaNjavora state only. Especially Syamasyastri stayed in TaNjavora city only.

Box6.3: English Translation of Kannada Text given in box 6.1

6.3.2 Tokenization

The transliterated text is tokenized. The tokenization process divides the text into the sequence of tokens or words. The delimiters like full stop (.), exclamatory mark (!), question mark (?), colon (:), comma (,) and semicolon (;) symbols present in Kannada do not contribute to named entities. We have also formulated rules to identify abbreviations in the text. If the sentence boundary delimiter comes with this abbreviation, it may or may not be the exact sentence boundary. The transliterated corpus after tokenization is shown in box 6.4.

ivattu / navaraatriya / muuraneya / dina / maisuuru / mattu / taMjaavuuru / eraDuu / 18-19 / ne / s`atamaanagaLalli / dakshiNa / bhaaratada / pramukha / saaMskRtika / nelegaLaagi / ruupugoMDavu / Haagaagiyee / iMdiguu / naavu / viiNe / citrakale / mattu / bharatanaaTya / iveraDaralluu / maisuuru / s`aili / mattu / taMjaavuuru / s`aili / eMdu / eraDu / pramukha / s`ailegaLannu / nooDabahudu / karnaaTaka / saMgiita / trimuurtigaLeMdee / hesaraada / tyaagaraaja / muttusvaami / diikshita / mattu / s`aamaas`aastri / avaru / baaLiddu / taMjaavuuru / raajyadallee / adaralliyuu / s`aamaas`aastri / avaru / iddaddaMtuu / taMjaavuuru / nagaradallee

Box 6.4: Tokens in Transliteration Form.

6.3.3 NER Recognizer Module

The named entity recognizer takes an input like “raam” and output raam as person name. The system uses a proper noun dictionary and rules list , suffix, and prefix list, and gazetteer list and regular expression. Tags used in NER are listed in table 6.1.

Table 6.1. Tag Sets Used for Rule Based NER

TAG	DESCRIPTION	EXAMPLE
N-PRP-PRSN	Person Name	Ram
N-PRP-ORG	Organization Name	Basaveshwar sugars ltd.
N-PRP-LOC	Place Name	Bombay
N-PRP-NUM	Numeric Value	12,345
N-PRP-TIM	Time	14 june Monday
N-PRP-MEA	Measurement	10 kg.

6.3.3.1 Proper Noun Dictionary

Each token is searched in the dictionary, if it is found in the dictionary then corresponding tag from the dictionary is assigned to the token. If the word is not found in the dictionary then it is passed to the NER module for further processing. We have developed a proper noun electronic dictionary of 5000 plus words manually for this work. Sample proper noun dictionary entries are shown in box 6.5.

kalakatta||N-PRP-LOC

baMbe||N-PRP-LOC

viveekaanaMda||N-PRP-LOC

raamu||N-PRP-PER

Box 6.5 Sample Proper Noun Dictionary

6.3.3.2 Proposed Rule Based Algorithm for Kannada NER

An algorithm for Kannada named entities is proposed by considering all the features mentioned the section 6.2. is given below.

- **Algorithm: Named Entity Recognition and Classification**

```
Input:  Transliterated Kannada Text
Output: Output text tagged with Named Entities tag.
```

Description: W_i-Current word, W_{i+1} -next word, W $_{i+2}$-next (next (word), W_{i-1}-previous word and soon. The named entities are Person, Organization, Location, Time, and Measurement

Start

Step 1: Read input File. Preprocess of text by splitting on space, removing unwanted characters like "? -|:" Etc., and if blank line does not process it.

Step 2: While (not end- of file) do

R1: IF (Wi is a number)AND (Wi+1 is a Unit of measure)
THEN (Wi, Wi+1) bigram is NE-Measurement

R2: IF (Wi is a number) AND (Wi+1 is an am|pm Unit)
THEN (Wi, Wi+1 bigram) is a NE-time

R3: IF (Wi is a number) AND (Wi+1 is the month's name) AND (Wi+2 is digit no.) THEN (Wi, Wi+1, Wi+2) trigram is NE-Time

R4: IF (Wi is a number) AND (Wi+1 is a month name) AND Wi+2 is not a four digit no) THEN (Wi, Wi+1) bigram is a Time NE

R5: IF (Wi denotes day) AND (Wi+1 is a number) AND (Wi+2 is month name) THEN (Wi, Wi+1, Wi+2) trigram is NE-Time

R6: IF(Wi is day) AND (Wi+1 is a number) AND(Wi+2 is not month name) THEN (Wi,Wi+1,) bigram is a NE-time

R7: IF(Wi is day) AND (Wi+1 is not number) AND (Wi+2 is not month name) THEN (Wi unigram is a NE-time)

```
R8: IF(Wi  matches with   month list) THEN Wi is  NE-Time

R9: IF (Wi matches date patterns)THEN(Wi unigram is a
time NE

R10: IF (Wi denotes prefix in the Designation list)
     THEN (Wi, Wi+1) bigram is NE-person

R11: IF(Wi ends with suffix in the person clue list) THEN
Wi
     is NE-person
R12: IF (Wi ends with suffix in the location list) THEN
Wi
     NE- location
R13: IF(Wi denotes suffix organization list)
      THEN (Wi, Wi-1, Wi-2) trigram NE-organization
R14: IF (Wi  matches  with suffix location street endings
     list) THEN (Wi,Wi-1) bigram is a NE-location
R15: IF (Wi mathces the pattern character, followed by
dot
     followed  by number) THEN Wi is NE-location or NE-
Time

#Search in Proper noun Dictionary
#Associative array to store Words Entries in Dictionary
tags;
R16: IF (Search Wi in the dictionary) THEN (Wi, Wi+1)
bigram as NE-Location or NE-Organization. Otherwise  Wi
is NE-tag
     in dictionary
Step 3 : Until End of input file
Stop
```

6.4 Results of Rule Based Approach

To evaluate the NER system we randomly downloaded corpus from the famous Kannada Newspaper PrajavaaNi website around 20 files table 6.3 of varying length and 6537 words and ran our NER through it, we observed that around 79.52% NEs are identified correctly, 12.34% are identified wrong and around 8.6% are left unrecognized. Precision is the percentage of correct positive predictions returned by the system. It is computed as the ratio between the number of NEs correctly identified by the system as True Positives (TP) and the total number of NEs returned by the system. The precision is calculated by dividing TP by the sum of TP and false positives (FP).

$$\text{Precision}=\frac{TP}{(TP+FP)} \qquad \ldots(6.1)$$

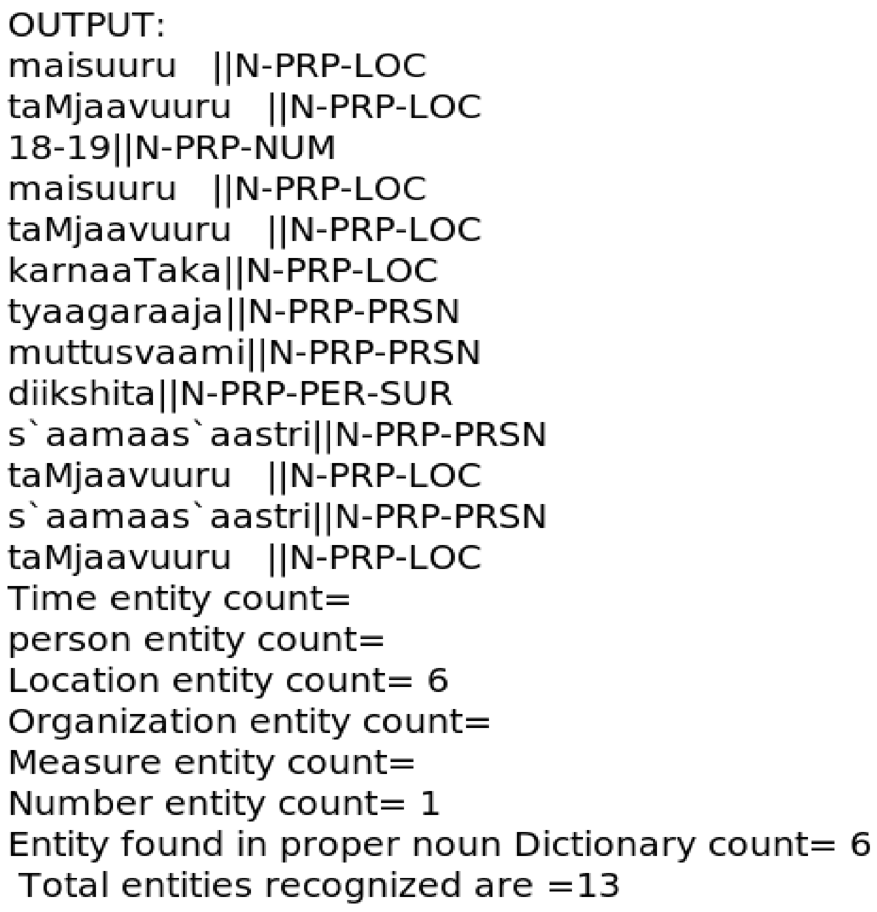
TP= 567, TP+FP=567+88. So Precision is 86%. Recall indicates the percentage of positive cases recognized by the system. It is computed as the ratio between the number of NEs correctly identified by the system (TP) and the number of NEs that the system was expected to recognize. Thus, Recall is the number of (TP) divided by the sum of (TP) and false negatives (FN).

$$\text{Recall}= \frac{TP}{(TP+FN)} \qquad \ldots(6.2)$$

$$\text{F-Measure} = \frac{2*Precision*Recall}{Precision+Recall} \qquad \ldots(6.3)$$

TP=567, TP+FN=567+61. Recall is 90%. F-measure is the common weighted harmonic mean between Precision and Recall defined as where beta is the weighting factor.

Rule based approach gives better results, but the amount of time required to design the rules is more. But the challenge lies in the development of rule- based techniques that require huge experience and grammatical knowledge of the language of interest. Experiments are carried out on the renowned Kannada PrajavaaNi news paper corpus. The sample output of NER for the text shown in figure 6.2 is shown in table 6.5. 3 location entities 4 person entities and 1 number entity are recognized.



```
OUTPUT:
maisuuru   ||N-PRP-LOC
taMjaavuuru   ||N-PRP-LOC
18-19||N-PRP-NUM
maisuuru   ||N-PRP-LOC
taMjaavuuru   ||N-PRP-LOC
karnaaTaka||N-PRP-LOC
tyaagaraaja||N-PRP-PRSN
muttusvaami||N-PRP-PRSN
diikshita||N-PRP-PER-SUR
s`aamaas`aastri||N-PRP-PRSN
taMjaavuuru   ||N-PRP-LOC
s`aamaas`aastri||N-PRP-PRSN
taMjaavuuru   ||N-PRP-LOC
Time entity count=
person entity count=
Location entity count= 6
Organization entity count=
Measure entity count=
Number entity count= 1
Entity found in proper noun Dictionary count= 6
 Total entities recognized are =13
```

Figure 6.2. Output of Rule Based NER

Table 6.2. Sample Output of Text given in Box 6. 2

S.No	English Word	Transliterated Word	Tag
1	Mysore	Maisuuru	N-PRP-LOC
2	Tanjavore	taMjaavuuru	N-PRP-LOC
3	18-19	18-19	N-PRP-NUM
4	Karnataka	karnaaTaka	N-PRP-LOC
5	Tyagaraj	Tyaagaraaja	N-PRP-PRSN
6	Syamashastri	Syamasyaastri	N-PRP-PRSN
7	Muttuswami	Muttuswami	N-PRP-PRSN
8	Dixit	Dikshit	N-PRP-PRSN

Table 6.3: Sample PrajaavaaNi News paper Files

File Names	Size of File in words	Wrong NES	Correct NE	Unrecognized	Total NEs
praja-29-09-2.txt.out	253	2	23	2	27
praja-29-09-3.txt.out	392	16	48	7	71
praja-29-09-4.txt.out	253	1	12	1	14
praja-29-09-5.txt.out	251	3	21	2	26
praja-29-09-6.txt.out	694	7	32	6	45
praja-29-09-7.txt.out	421	1	16	4	21
praja-29-09-8.txt.out	191	2	10	2	14
praja-29-09-9.txt.out	912	3	8	2	13
praja-29-09-11.txt.out	261	1	6	1	8
praja-29-09-12.txt.out	784	14	54	2	70
praja-29-09-13.txt.out	215	4	22	4	30
praja-5-6-12-1.txt.out	701	8	71	0	79
praja-5-6-12-2.txt.out	532	3	32	3	38
praja-5-6-12-	537	3	22	2	27

3.txt.out					
praja-4-6-12.txt.out	413	2	26	7	35
praja-4-6-12-1.txt.out	179	2	19	3	21
praja-4-6-12-3.txt.out	292	3	24	2	29
praja-4-6-12-4.txt.out	409	7	63	9	79
praja-4-6-12-5.txt.out	360	6	58	2	66
Total		88	567	61	713
Average		4.4	28.35	3.05	35.65

A confusion matrix contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix. The table 6.5 shows the confusion matrix for the Prajaavani corpus files. The evaluation of NER output is shown in table 6.4.

Table 6.4. Evaluation of NER Output

Number of Total NEs in Test Data	713
Number of NEs identified	655
Number of Correct identified NEs	567
Precision	86%
Recall	90%
F-measure	87.95%

Table 6. 5. Results on Corpus of Different Size

Corpus Size in Words	TP	FP	FN	TN	Precision	Recall	F-measure
2423	136	29	18	2269	82.42%	88%	85%
4203	230	50	29	3944	83.42%	89%	86%
6537	403	70	41	6093	85.14%	90.7%	87%

The precision and recall are shown figure 6.3. The total recognition rate for first 19 files with rate of identification of correct Nes, wrong Nes and unrecognized Nes is shown in figure 6.4

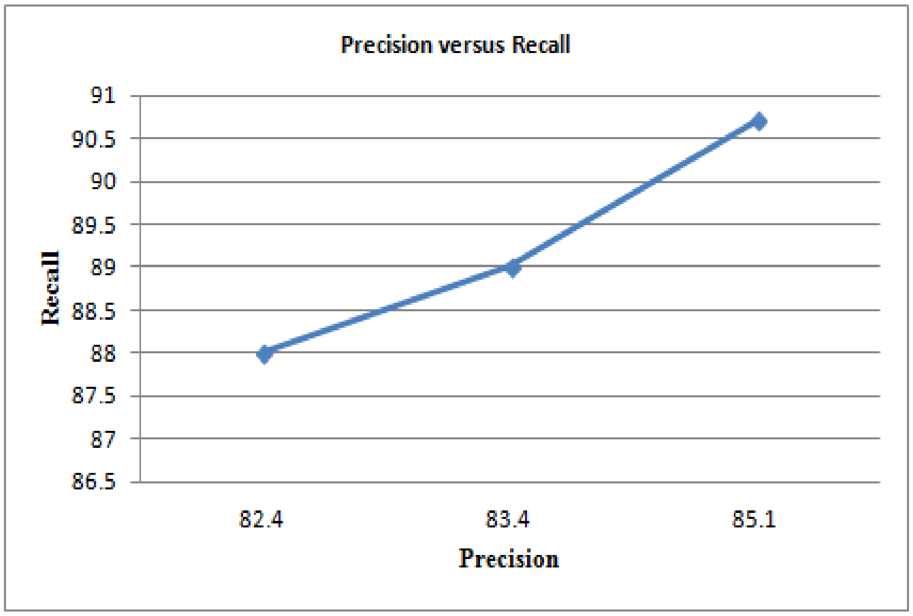


Figure 6.3. Precision versus Recall

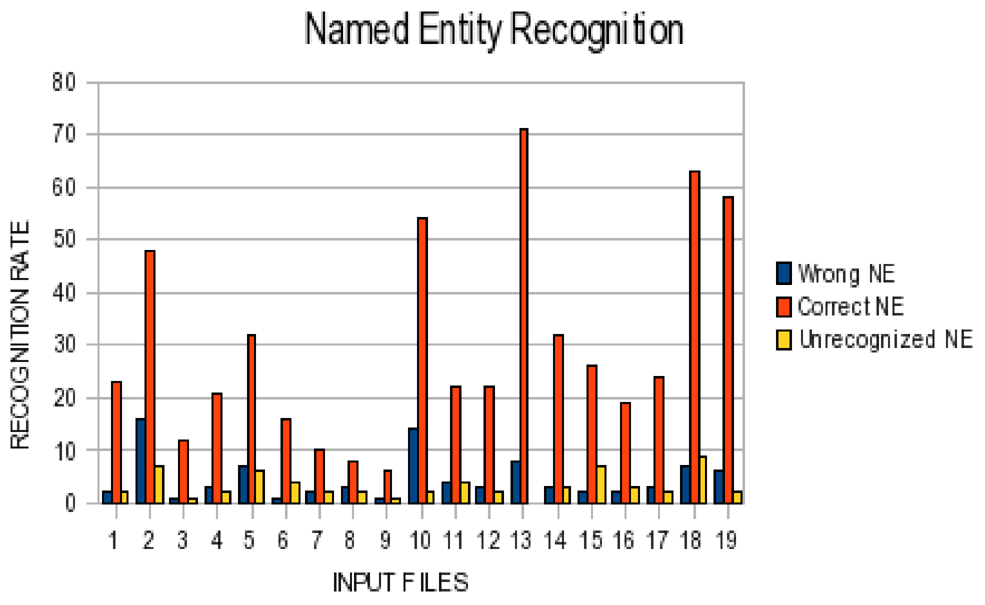


Figure 6.4. Recognition in Rate in Rule Based Approach

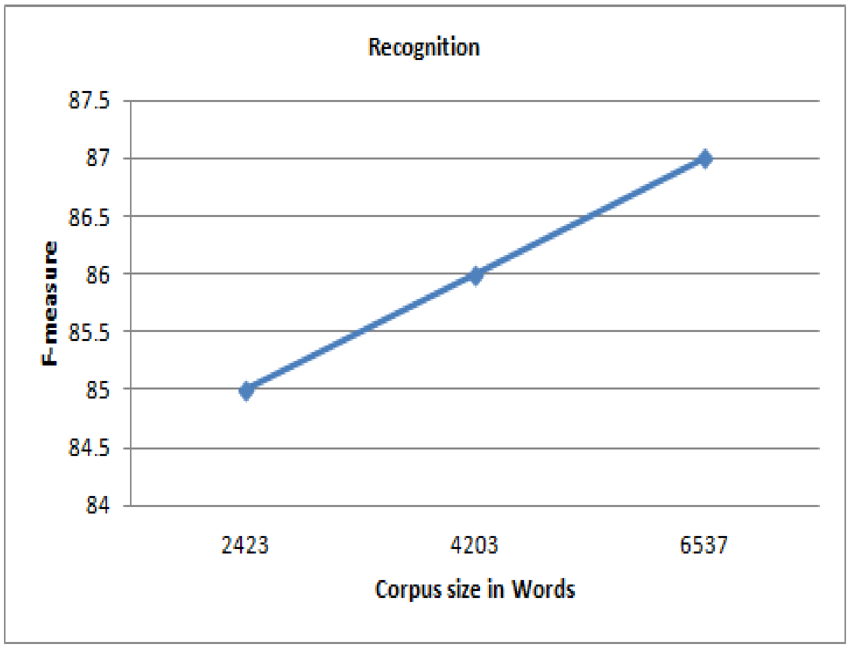


Figure 6.5. Corpus size versus F-Measure

Precision versus Recall graph figure 6.3 shows that our precision and recall values do not differ much. In a recognition task, the precision for a class is the *number of true positives* (i.e. the *number of items correctly labeled as belonging to the positive class*) *divided by the total number of elements labeled as belonging to the positive class* (i.e. the sum of true positives and false positives which are items incorrectly labeled as belonging to the class). Recall in this context is defined as the *number of true positives divided by the total number of elements that actually belong to the positive class.* Figure 6.4 is a Recognition graph which shows a plot with F-measure on the Y axis and the corpus size in words on the X-axis is shown in figure 6.5. **F-measure**, which is the *weighted harmonic mean of precision and recall*. We observe that there is good recognition rate and from the graph 6.5 it is observed that there is an increasing rate of recognition with increase in corpus size in terms of number of words, but however this is not always the case, it depends on the corpus we are choosing and distribution of NEs in the corpus.

• Summary

NER for Kannada is a challenging task due to its inherent ambiguity nature and lack of capitalization feature. The attempted work is the first of its kind for Kannada. The developed methodology for Kannada named entities has given good recognition rate and precision around 86%. Performance can be improved by deploying gazetteer lists like proper noun dictionary with prefixes and suffixes. Proposed rule based ner is language independent and the Kannada languages aspects are not hard coded in the program. Hence can be used for other languages just by replacing the language aspect file with their language.

Chapter 7

Named Entity Recognition and Classification Using Hidden Markov Model

In the previous chapter a rule based methodology to recognize Kannada named entities is discussed. In this chapter Hidden Markov Model (HMM) as the statistical machine learning approach to recognize Kannada named entities is discussed. The main idea behind the use of HMM model for building NER system is that it is language independent and can be applied system for any language domain. ***In this work the states are not fixed, meaning it is dynamic in nature. One can use it according to their interest and add later any states with change of domain***. The corpus used here is also not domain specific. *HMM is implemented using Java, JDK 1.2 NET BEANS IDE, in Windows.* ***There is also flexibility provided in giving input to the system, the input may be text file, doc, docx and also pdf document.*** **Grapgviz software is used to show** graphically the state transition probabilities and other probabilities. The accuracy is around 93%.

7.1 Introduction to Hidden Markov Model

A **Hidden Markov Model** (**HMM**) is a statistical model, the system is modeled as the Markov process with *hidden* states. In a simpler Markov model the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In a hidden Markov model, the state is not directly visible but hidden and output dependent on the state is visible. Each state has a probability distribution over the possible output tokens. Therefore the chain of tokens generated by the HMM gives information about the sequence of states. The Hidden Markov Model (HMM) is a generative model. The joint probability of observation and label sequence is assigned by model. Here the parameters are trained to maximize the joint likelihood of the training sets. It is advantageous as its basic theory is elegant and easy to understand. Hence it is easier to implement and analyze. It uses only positive data, so they can be easily scaled (B.D. Scott, 1997) (Sudha Morwal, Nusrat jahan, 2012).

NER and NLP research around the world has taken giant leaps in the last twenty two years. NER field has a history of more than twenty two years of research from 1991 to 2014 with the advent of effective machine learning algorithms and the creation of large annotated corpora for various languages. The survey of NER and various approaches followed till today in English and other foreign languages and Indian languages show difficulties of NER with respect to Kannada. In English language, lots of work has been done in this field, where capitalization is a major clue for making rules. The HMM technique produces optimal state sequences. It is based on Markov Chain Property i.e. the probability of occurrence of the next state is dependent on the just previous state as shown in Figure7.1.

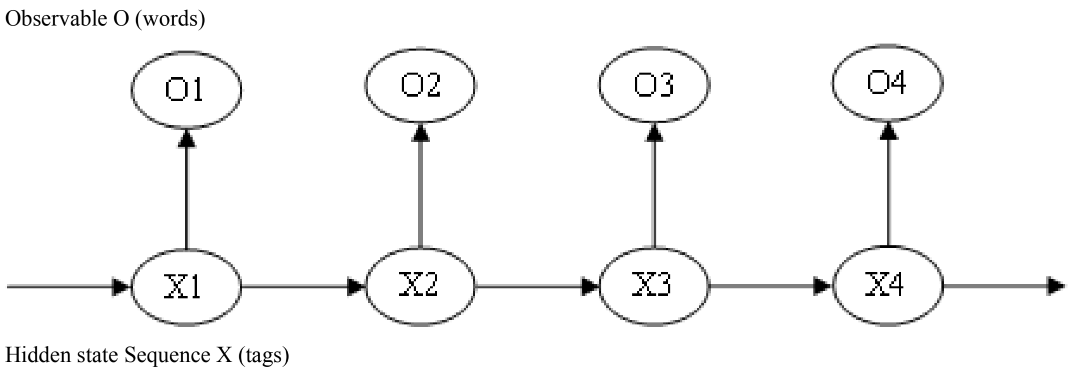


Figure 7.1. Diagrammatic description of Hidden Markov Chain Property

7.1.1 Notations used in HMM:

Hidden Markov Model can be expressed mathically with the following notations.

T = the length of the observation sequence (Number of words in the observation text).

N = the number of states in the model (Number of Named Entities) (N=9 maximum).

M = the number of observation symbols (Number of words in the tagged text corpora).

$S= \{S_1, S_2, S_{N-1}\}$ (The hidden states of the Markov process).

$V= \{0, 1 \ldots\ldots, M-1\}$ (Set of possible observations).

A = The state transition probability matrix (Size NxN maximum).

B = The observation probability matrix (or emission probability matrix NxM).

π = The initial state distribution probabilities (1xN) .

O= (O_1, O_2, O $_{T-1}$) is Observation sequence.

HMM is denoted by **λ = (A, B, π)**. Where, **'A'** represents the transition probability. **'B'** represents emission probability and **'π'** represents the starting probability. Each Hidden Markov Model is defined by states, state probabilities, transition probabilities, emission probabilities and initial probabilities. In order to define an HMM completely, the following five Elements have to be defined.

1. The N states of the Model are defined by

$S= \{S_1, S_{N-1}\}$

2. The M observation symbols per state are $V= \{V_1, VM\}$. If the observations are continuous then is M infinite.

3. The state transition probability distribution is given as $A= \{a_{ij}\}$, where a_{ij} is the transition probability, that the state at time $t+1$ is S_j, when the state at time t is S_i . The structure of this stochastic matrix defines the connection structure of the model. If the coefficient a_{ij} is zero, it will remain zero throughout the training process, so there will never be a transition from state S_i to S_j.

$$a_{ij}=p\{q_{t+1}=j|q_t=i\},1\leq i,j\leq N \qquad \text{...(7.1)}$$

Where q_t denotes the current state. The transition probabilities should satisfy the normal stochastic constraints, $a_{ij}\geq 0,\ 1\leq i, j \leq N$ and $\sum_{j=1}^{N} a_{ij}=1,\ 1\leq i \leq N$

4. The Observation symbol for probability distribution in each state is, $B= \{b_j(k)\}$, where $b_j(k)$ is the probability that symbol V_k is emitted in state Sj.

$$b_j(k) = p\{o_t = V_k | q_t = j\},\ 1 \le j \le N,\ 1 \le k \le M \quad \text{...(7.2)}$$

Where V_k denotes the k_{th} observation symbol in the alphabet, and O_t the current parameter vector.

The following stochastic constraints must be satisfied:

$$b_j(k) \ge 0,\ 1 \le j \le N,\ 1 \le K \le M \text{ And } \sum_{k=1}^{M} b_j(k) = 1,\ 1 \le j \le N \quad \text{...(7.3)}$$

5. The HMM initial state distribution $\pi = \{\pi_i\}$, where π_i is the probability that the model is in state S_i at the time t=0 with

$$\pi_i = p\{q_1 = i\} \text{ and } 1 \le i \le N \quad \text{...(7.4)}$$

While defining the HMM, it is also very important to clarify whether the model will be discrete, continuing or a mixed form.

7.1 .2 Three Basic Problems of HMMs

In the following part we will first analyze the three well-known basic problems of Hidden Markov Models.

7.1.2.1 The Evaluation Problem and the Forward Algorithm

Given a model $\lambda = (A, B, \pi)$ a sequence of observations $O = o_1, o_2, o_T$, $p\{o/\lambda\}$ needs to be found. Although this can be calculated by the use of simple probabilistic arguments, it is not very practicable because the calculation involves a number of operations in the order of N ^{T}. But fortunately there is another calculation method with considerably low complexity that uses an auxiliary variable like below.

$$\alpha_t(i) = p\{o_1, \dots, o_{t,}, q_t = i | \lambda\} \quad \text{...(7.5)}$$

$\alpha_t(i)$ is called forward variable , and o_1, o_2, o_T are the partial observation sequences. Out of this the recursive relationship is as below.

$$\alpha_{t+1}(j) = b_j(o_{t+1}) \sum_{i=1}^{N} \alpha(i) o_{ij},\ 1 \le j \le N,\ 1 \le t \le T-1 \quad \text{...(7.6)}$$

With $\alpha_1(j) = \pi_j b_j(o_1), 1 \le j \le N$ follows. $\alpha_T(i), 1 \le j \le N$ can be calculated using recursion. So the required probability is given by

$$p\{O|\lambda\} = \sum_{i=1}^{N} \alpha_T(i) \qquad \text{...(7.7)}$$

This method is commonly known as forward algorithm. The backward variable $\beta_t(i)$ can be defined similarly.

$$\beta_t(i) = p\{o_{t+1}, o_{t+2}, \dots\dots o_t | q_t = i, \lambda\} \qquad \text{...(7.8)}$$

Given that the current state i, $\beta_t(i)$ is the probability of the partial observation sequence $o_{t+1}, o_{t+2}, \dots\dots o_T$. $\beta_t(i)$ can also be calculated efficiently by using a recursion.

$$\beta t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(o_{t+1}), 1 \le i \le N, 1 \le t \le T-1 \qquad \text{...(7.9)}$$

Where $\beta_T(i) = 1, 1 \le i \le N$. Further it is seen that

$$\alpha_t(i)\ \beta_t(i) = p(O, q_t = i|\lambda\}, 1 \le I \le N, 1 \le t \le T \text{...(7.10)}$$

So there are two ways to calculate $p\{O|\lambda\}$, by using either or backward variable:

$$p\{O|\lambda\} = \sum_{i=1}^{N} o, q_t = i|\lambda = \sum_{i=1}^{N} \alpha_t(i)\, \beta_t(i) \qquad \text{...(7.11)}$$

This equation can be very useful, especially in deriving the formulas required for gradient based training.

7.1.2.2 The Decoding Problem and the Viterbi Algorithm

Given the observation sequence $O=0_1, 0_2, \dots, o_T$ and a model $\lambda=(A, B, \pi)$, A search *for the most likely state sequence is made. The "likely state sequence" definition influences the* solution to the problem. In one approach, it is necessary to find the most likely state q_t and to concatenate all such 'q_t.' But this approach sometimes does not result in a meaningful state sequence, another method commonly known as Viterbi algorithm can be used as alternative. Using the Viterbi algorithm, the whole state sequence with maximum likelihood can be found.

An auxiliary variable which gives the highest probability can be defined such that the partial observation sequence and state sequence up to $t=t$, given the current state is i is as follows.

$$\delta t(i)=\max_{q1,q2,\ldots,qt-1} p\{q_1, q_2, \ldots, q_{t-1}, q_t = i, o_1, o_2, \ldots, o_{t-1} \mid \lambda\} \quad \ldots\textbf{(7.12)}$$

It follows that

$$\delta_{t+1}(j)= b_j(o_{t+1})[\max_{0 \le i \le N} \delta_t(i)\, a_{ij}],\ 1 \le i \le N,\ 1 \le t \le T-1 \quad \ldots\textbf{(7.13)}$$

With $\delta_1(j)=\pi_j b_j(o_1),\ 1 \le j \le N$

So we start from the calculation of $\delta_T(j),\ 1 \le j \le N$ to calculate the most likely state sequence. We always keep a pointer to the"winning state" in the maximum finding operation. It results in state j^*.

where $j^* = arg \max_{1 \le j \le N} \delta_T(j)$.

We start from this state and back-track the sequence of states as the pointer in each state indicates. So we get the required set of states. This whole algorithm can be interpreted as a search in a graph whose nodes are formed by the states of the HMM in each of the time instant $t,\ 1 \le t \le T$.

7.1.2.3 The Learning Problem

How to adjust the parameters of HMM in a way that a given set of observations (the *training set*) is represented by the model in the best way for the intended application?. Based on the application, the "quantity" that is to be optimized during the learning process is different. Hence there are many optimization criteria for learning. Generally used two main optimization criteria are Maximum Likelihood (ML) and Maximum Mutual Information (MMI).

7.2 Types of Hidden Markov Models

Depending on problem complexities, signal processing applications and requiremnet, it is indispensable to choose the appropriate type of HMM very early in the concept and design a phase of modern HMM based systems. Different types of HMMs are introduced and some generalized criteria is designed to choose the right type of HMM in order to solve different kinds of problems.

7.2.1 Discrete HMM

- *Problematic:* assuming that we have continuous valued feature vectors we will summarize in this section how to use Discrete Hidden Markov Models (DHMM) to solve the problem.

- *Generalized Methodology:* the following three steps have to be processed:

1. A set of d-dimensional real valued vectors should be reduced to k d-dimensional vectors → vector quantization by code book (k-means cluster algorithm)
2. Find the nearest codebook vector for the current feature vector
3. Use the index of this codebook vector for DHMM emission symbol / input

7.2.2 Continuous HMM

It is assumed that the output can be written as

$$b_t(x) = \sum_{k=1}^{k} c_{jk} N(x|\theta_{jk}) \quad \text{...(7.14)}$$

With $\sum_{k=1}^{k} c_{jk} = 1$, where c_{jk} is the mixture coefficient and $N(x|\theta_{jk})$ is the Gaussian density. For each state K multivariate Gaussian densities and K mixture coefficients have to be estimated. This results in the following parameters for each state: covariance matrix, mean vector and mixture coefficients vector.

A Continuous Hidden Markov Model (CHMM) is a three-layered stochastic process. The first part is equal to DHMM, the selection of the next state. The second and the third part are similar to the selection of emission symbol with DHMM, whereas the second part of CHMM is the selection of the mixture density by mixture coefficient. The selection of the output symbol by the Gaussian density is the third and last part.

The training algorithms and classification are to be modified. There are only minor changes in the classification algorithm. The modified probability densities have to be substituted. The Baum-Welch Viterbi training algorithms have to be modified by additional calculation. The disadvantage is a high computational effort. The Gaussian distributions have to be evaluated and the high number of parameters probably results in instabilities.

7.2.3 Semi-Continuous HMM

The Semi-Continuous HMM (SCHMM) can be seen as a compromise between DHMM and CHMM. It is assumed that the output can be written as

$$b_t(x)=\sum_{k=1}^{k} c_{jk} P(x|\theta_k) \qquad \text{...(7.15)}$$

With $\sum_{k=1}^{k} c_{jk}$ where c_{jk} is the mixture coefficient and $P(x|\theta_k)$ the Gaussian distribution. Overall, K multivariate Gaussian distributions and K mixture coefficients have to be estimated. In contrast to the CHMM, same set of Gaussian mixture densities are used for all states. Like the CHMM, the SCHMM is a three-layered stochastic process. After the next state has been selected, there will be the selection of the mixture density by the mixture coefficient. Third, the output symbol (vector) has to be selected by Gaussian density. The second and third step is similar to the selection of emission symbol with DHMM. There are some modifications regarding classification and training algorithms.

For classification algorithm, the modified probability densities have to be modified and the Baum-Welch/Viterbi training algorithm are modified by additional calculations. The disadvantage is a high computational effort. The Gaussian distributions have to be evaluated and the high number of parameters may result in instabilities. Altogether, the modifications made are similar to those in the CHMM, but the number of parameters is reduced significantly.

7.3 Features of the Proposed HMM Based NER System

- *Language independent:* This methodology works for any natural language, European language also. This work is tested for Kannada language.
- *General Approach:* This approach is not domain specific. This work is tested for news paper corpus can also be tested for tourism corpus, general sentences and stories.
- *High Accuracy:* If the rich corpus is developed, it performs better. During testing, accuracy up to 90% is obtained.
- *Usefulness to other classification:* Since the parameters are dynamic in nature the same NER system can be used for other NLP classifications like Part-of-speech tagging etc.
- *Use of Annotated corpus:* To use this system tagged is to be developed either with the help of the proposed system or with other tools. This tagged corpus can be used in other natural language processing applications.

7.4. Advantages of HMM

The HMM based NER system is easily understandable and is easy to implement and analyze.

- It can be used for any amount of data, so the system is scalable.
- We can develop NER system which is language independent. We can use it for any language domain.
- It solves the sequence labeling problem very efficiently.
- The states used in the model are also not fixed. One can use it according to their requirements or interest, meaning it is dynamic in nature.

7.5 Disadvantages of HMM

- In order to define joint probability over observation and label sequence, HMM needs to enumerate all possible observation sequences. Hence it makes various assumptions about data like Markov chain assumption i.e. current label depends only on the previous label.

- It is not practical for representing multiple overlapping features and long term dependencies.
- Number of parameters to be evaluated is huge. So it needs a large data set for training.

In the next section the proposed system using Hidden Markov Model (HMM) is described. HMM is used to compute probability of occurrence of the named entities in the corpus i.e. in the given text document. Viterbi algorithm is used to find the most probable state sequences. The **Viterbi algorithm** is a dynamic programming algorithm for finding the most likely sequence of hidden states – called the **Viterbi path** which results in a sequence of observed events. Figure 7.2 shows the proposed system.

7.6 Proposed Named Entity Recognition System Using HMM

Our objective is to build a machine learning named entity recognition system, which when given a new previously unseen text can identify and classify the named entities in the text. This means that your system should annotate each word in the text with one of the nine possible classes. We are using the Hidden Markov Model based machine learning approach. HMM is a statistical based approach in which states are hidden or unobservable. Here, the state is not directly visible, but the output, which depends on the state, is visible. In Hidden Markov the joint distribution of observations and hidden states, or equivalently both the prior distribution of hidden states (the *transition probabilities*) and the conditional distribution of observations of given state (the *emission probabilities*), are modeled. In this proposed NER system we are using 9 NE tags, namely NEP, NEL, NEO, NED,NETE, NEM,NET etc. as shown in Table 7.1

The system performs in three phases; the first phase is referred to as 'Annotation phase' that produces tagged or annotated document from the given raw text. The second phase is referred to as 'Training phase'. In this phase, it computes three most essential parameters of HMM i.e. Start Probability (π), Transition Probability (B), and Emission Probability (A). The last phase is referred to as 'Testing phase'. In this phase, the user gives certain test sentences to the system, and based on the HMM parameters computed in the previous state, Viterbi algorithm computes the optimal state sequences for the given sentence. The overall logical flow of untagged raw text to tagged text is depicted in Figure 7.2.

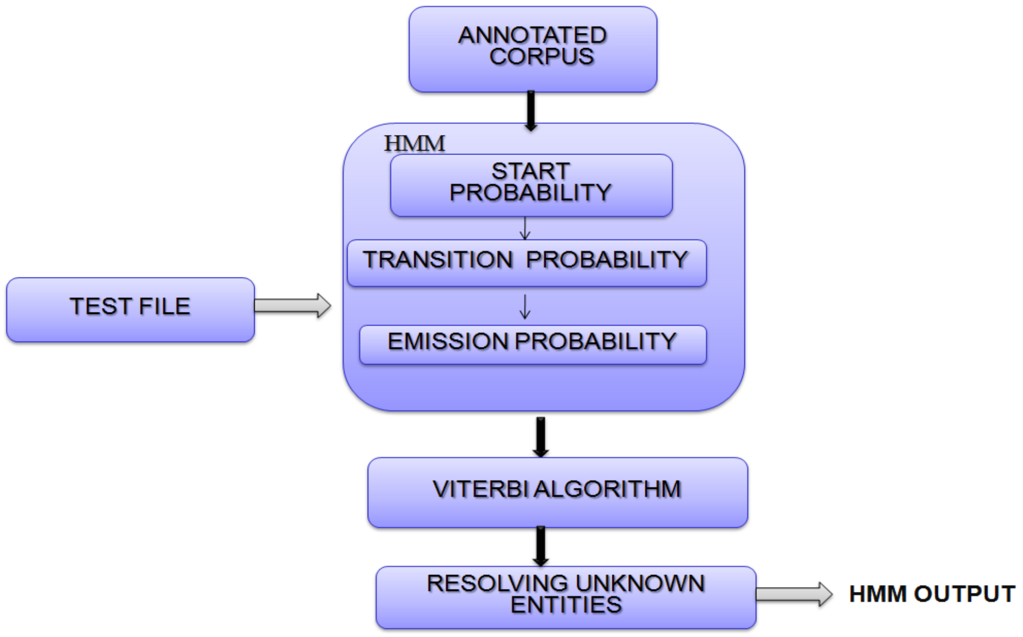


Figure 7.2 Diagrammatic description of NER using HMM

7.6.1 Annotation Phase

The raw data is converted into trainable form, so as to make it suitable to be used in the Hidden Markov model framework of Kannada language. The training data may be collected from any source like the Kannada website, Kannada News Papers or from any open source, tourism corpus or simply a plaintext file containing some sentences. So in order to put these files in trainable form we have to perform the following steps as shown in algorithm 7.1.

7.6.1.1: Algorithm for Annotation Phase

```
Input : Raw text file
Output: Annotated Text (tagged text)
Procedure:
Step1: Separate each sentence from the raw text file.
Step2: Tokenize the words.
Step3: Tag (Named Entity tag) the words by the user
experience.
Step4: Now the corpus is in trainable form.
```

Table 7.1. Named Entity tags used for HMM

S.No.	NE tags	Meaning	Example
1	NEP	Name of a person	allamprabhu, chinmaya, basu
2	NEL	Name of a location	bijapur, cenai
3	NEO	Name of an organization	infosis, aabiyam
4	NET	Measure of time	9.30 pm, 1-1-2014, Monday, 14th june etc
5	NED	Name of a month	Jaanevary, juun
6	NETE	Name clue	siiem,
7	OTHER	Not a named entity	-
8	NEM	Any quantity like kg	10 kg, 1itre
9	NEN	Number	12, 334 etc

7.6.2 Training Phase

In training phase three parameters of the HMM are computed:

- *Procedure to find Start probability:* Start probability gives the probability that the sentence starts with a particular tag. The start probability computes the sentences starting with the word containing a particular tag using algorithm 7.2. The start probability is computed in Table 7.2.
-

7.6.2.1 Algorithm for Start Probability

```
Input : Annotated Text file
Output: Start Probability
Procedure:
Step1: For each starting tag
Step2: Find frequency of that tag as starting tag
Step3: Calculate π
```

$$\text{Start Probability}(\pi) = \frac{(\text{Number of Sentences starting with a particular tag})}{(\text{Total Number of Sentences})} \quad \ldots \ (7.16)$$

- *Procedure to find Transition probability:* The state transition probabilities A = $\{a_{ij}\}$, is the probability of the state moving from i to j in one transition. In part-of-speech tagging the states correspond to tags, so a_{ij} is the probability that the model will move from tag t_i to t_j (where t_i, $t_j \in \{T\}$). In other words, a_{ij} is the probability that t_j follows t_i (i.e. $P(t_j \mid t_i)$). This probability is usually estimated from the annotated training corpus during training. Using algorithm 7.3.Where the transition probability is computed in Table 7.3.

7.6.2.2 Algorithm for Transition Probability

```
Input : Annotated text file
Output: Transition Probability
Procedure:
Step 1: For each tag in states (Ti)
Step 2: For each other tag in states (Tj)
Step 3: If Ti not equal to Tj
Step 4: Find frequency of tag sequence Ti Tj i.e. Tj
after Ti
Step 5: Calculate A = frequency (Ti Tj) / frequency (Ti)
```

$$\text{Transition Probability (A)} = \frac{\text{Total Number of sequences from Ti to Tj}}{\text{(Total Number of Ti)}} \quad \ldots (7.17)$$

- *Procedure to find Emission probability:* Emission probability is the probability of assigning a particular tag to the word in the corpus or document. The emission probability computes the word having a particular tag in the whole corpus or gazetteer using algorithm 7.4. The emission probability is computed in Table 7.4.

7.6.2.3 Algorithm for Emission Probability

```
Input : Annotated Text file
Output: Emission Probability
Procedure:
Step 1: For each unique word Wi in annotated corpus
Step 2: Find frequency of word Wi as a particular tag Ti
Step 3: Divide frequency by frequency of that tag Ti
```

$$\text{Emission Probability(B)} = \frac{\text{(Total Number of occurrences of word as a tag)}}{\text{(Total occurrences of that tag)}} \quad \ldots (7.19)$$

7.6.3 Testing Phase

After calculating all these parameters apply the calculated parameters to the Viterbi algorithm and the sentences being tested as an observation to find named entities. The Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of hidden states called the Viterbi path – that results in a sequence of observed events, especially in the context of Hidden Markov Model. The below diagram describes the flow of the Viterbi algorithm Figure 7.3.

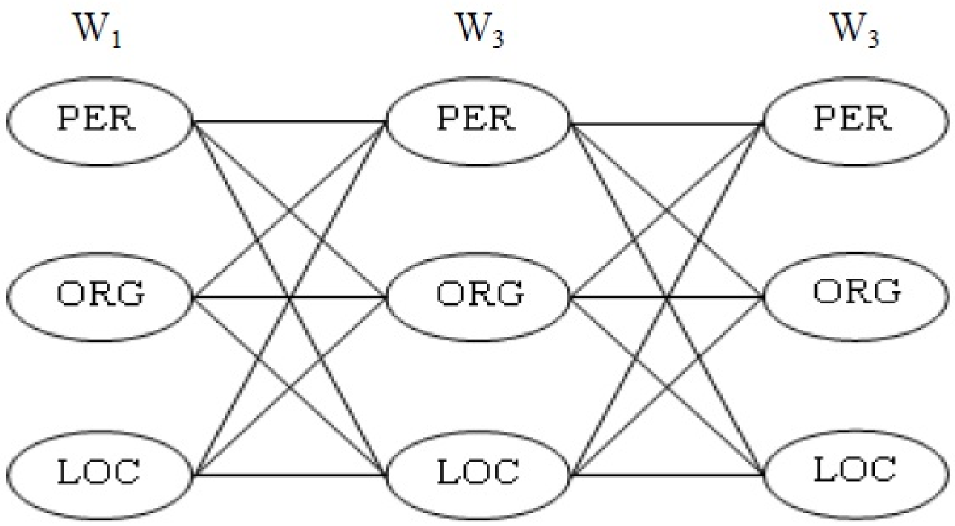


Figure7.3 Diagrammatic description of Viterbi algorithm

The algorithm consists of two phases. In the first phase we use simple dynamic programming technique to calculate the probability of the most probable state-path ending in each state. In the second phase we use data gathered in the first phase to find the most probable state-path. The Viterbi algorithms are interested only in the single most probable path, not in the sum of all paths generating a particular sequence. We calculate the value of *i(j)* as the maximum probability of the state path terminating in state *j*, using previous values of*i-1* . The value *i(j)* is therefore derived from exactly one of the previous values . To simplify the computation in the second phase, we will remember the state from which the value is derived. This is the second-last state of the most probable state path ending in state j. We finish the first phase of the algorithm by computing the probability of the most probable path s, which will be the maximum value in the vector.

In the second phase of the algorithm, which we call back-tracing, we reconstruct the sequence of states in the most probable state path. We do this, starting from the maximum value in the last vector and working the way back to the beginning of the sequence by following the back pointers which are stored as *i(j)*. This sequence of states (in reverse order) is output as the most probable state path.

7.6.3.1 Algorithm for Viterbi Path (M, X)

```
1: for l = 1 to m do {Initialization}
2: $\gamma_o(l)=\pi_j$
3: $\delta_o(l)=0$
4: end for
5: for j=1 to n do {first phase}
6: for k= 1 to m do
7: $\gamma_j (l)=\max_{k=1} m \gamma_{j-1} (k)t_k (i)e_l(X_j)${Recurrence}
8: $\delta_j (l)=\operatorname{argmax}_{k=1} m \gamma_{j-1} (k)t_k (l)e_i(X_j)$
9: end  for
10: end for
11:$S_n = \arg\max_{k=1} m \gamma_n (k)$
12: for l= n-1 to 1 do { Second phase ( back – tracing)}
13: $S_l =\delta_{l+1} (S_{l+1})$
14: end for
15: return S
```

Consider the raw text containing 5 sentences of Kannada language.

- s`ruti bi.el.di.i thaaMthrika mahaa vidhyaalaya, bijaapuradalli unnata vyaasaMga maaDuttiddaaLe.
- s`ruti em.Tek nalli kaMpyuTar sains iMjaniyariMg vibhaagavannu thegedhukoMDiddaaLe.
- tanna aMthima varshada projekTannu pro.ci.es.kusur ravara maargadhars`anadalli maaDuttiddaaLe.
- pro.ci.es.kusur bi.el.di.i thaaMthrika mahaa vidhyaalaya, bijaapuranalli upanyaasakaraagi kaarya nirvahisuttiddaare.
- s`ruti bijaapuradalli vaashisuttiddaaLe.

Now the annotated text is as follows:

- s`ruti/NEP bi.el.di.i thaaMthrika mahaa vidhyaalaya/NEO ,/OTHER bijaapuradalli/NEL unnata/OTHER vyaasaMga/OTHER maaDuttiddaaLe/OTHER ./OTHER
- s`ruti/NEP em.Tek/OTHER dalli/OTHER kaMpyuTar/OTHER sains/OTHER iMjaniyariMg/OTHER vibhaagavannu/OTHER thegedhukoMDiddaaLe/OTHER ./OTHER
- tanna/OTHER aMthima/OTHER varshada/OTHER projekTannu/OTHER pro.si.es.kusura/NEP ravara/OTHER maargadhars`anadalli/OTHER maaDuttidaaLe/OTHER ./OTHER
- pro.ci.es.kusura/NEP ravaru/OTHER bi.el.di.i thaaMthrika mahaa vidhyaalaya/NEO ,/OTHER bijaapuradalli/NEL upanyaasakaraagi/OTHER kaarya/OTHER nirvahisuttidaare/OTHER ./OTHER
- s`ruti/NEP bijaapuradalli/NEL vaasisuttiddaaLe/OTHER ./OTHER

In the above sentence, the NER based system identifies the NEs and classifies them into different NEs classes. Here, 's`ruti' and 'pro.ci.es.kusura' are the NEs and are Names of Persons. So these are allotted Name of Person tag (NEP). We have tagged 'bijaapuradalli' as NEL, since it is specified as the location name. The rest of the tokens are given OTHER tag since these are not Named Entities.

Now we calculate all the parameters of HMM model.

States= {NEP, NEL, NEO, OTHER}

Table 7.2. Start Probability π

NEP	NEL	NEO	OTHER
4/5	0/5	0/5	1/5

Table 7.3. Transition Probability A

	NEP	NEL	NEO	OTHER
NEP	0/5	1/5	1/5	3/5=0.60
NEL	0/3 =0.00	0/3= 0.00	0/3=0.00	3/3=1.00

NEO	0/2=0.00	0/2=0.00	0/2=0.00	2/2=1.00
OTHER	1/29=0.034	2/29=0.068	1/29=0.034	20/29=0.689

Table 7.4. Emission Probability B

	s`ruti	bijapuradalli	vaasisuttiddaaLe	
PER	3/5= 0.60	0/5=0.00	0/5=0.00	0/5=0.60
LOC	0/3 =0.00	3/3= 1.00	0/3=0.00	0/3=0.00
ORG	0/2=0.00	0/2=0.00	0/2=0.00	0/2=0.00
OTHER	0/29=0.00	0/29=0.00	1/29=0.034	5/29=0.17

7.7 Experimental Results

In this section a discussion about the training and testing data of the proposed system and some sample examples of Kannada is revealed. Transliteration program ur.pl is developed to convert Unicode text to Roman form.

7.7.1 Sample Training Data

Two sets of data namely training and testing are used. The raw data is converted into trainable form, so as to make it suitable to be used in the Hidden Markov model framework. First each sentence from the raw text file is tokenized assigned tag based on user experience. Now corpus is in trainable form that is shown in Table 7.5.

- *Observed data:* When training data is fully observed and there are no hidden variables, the only thing to be done is to count the frequencies and form the three matrices: π, A and B.
- *Unobserved data:* When there are hidden variables, there will be some free parameters which cannot be found by counting. At this moment, Expectation-Maximization (EM), or namely Baum-Welch, is used. In Baum-Welch algorithm there are two steps done on all instances. In each training iteration, the first step is to calculate the expectation of training instances and second is to maximize it.
- The Sample of Kannada text used in training and testing are listed below.

- **Example 1: Sample Kannada Raw Text**

ಜಾರ್ಖಂಡ್ನನೂತನ ಮುಖ್ಯಮಂತ್ರಿ ಅಭ್ಯರ್ಥಿಯನ್ನಾಗಿ ಮಾಜಿ ಡಿಸಿಎಂ ರಘುಬರ್ದಾಸ್ ಆಯ್ಕೆಯನ್ನುಇಂದು ಇಲ್ಲಿ ನಡೆದ ಶಾಸಕಾಂಗಸಭೆಯಲ್ಲಿ ಅಂತಿಮಗೊಳಿಸಲಾಗಿದೆಎಂದು ಬಿಜೆಪಿ ಮೂಲಗಳು ತಿಳಿಸಿವೆ. ಇಲ್ಲಿ ನಡೆದ ಬಿಜೆಪಿ ಶಾಸಕಾಂಗ ಪಕ್ಷದ ಸಭೆಯಲ್ಲಿ ದಾಸ್ ಅವರನ್ನು ಅಂತಿಮವಾಗಿ ಪಕ್ಷದ ಎಲ್ಲಾ ಶಾಸಕರ ಸಮ್ಮತಿಯೊಂದಿಗೆ ಮುಖ್ಯಮಂತ್ರಿ ಅಭ್ಯರ್ಥಿಯನ್ನಾಗಿ ಘೋಷಿಸಲಾಯಿತು ಎಂದು ತಿಳಿದುಬಂದಿದ್ದು, ಈ ಹಿಂದೆ ಕೇವಲ ಬುಡಕಟ್ಟು ಜನಾಂಗದವರೇ ಮುಖ್ಯಮಂತ್ರಿ ಸ್ಥಾನಕ್ಕೇರುತ್ತಿದ್ದ ಜಾರ್ಖಂಡ್‌ನಲ್ಲಿ ಇದೇ ಮೊದಲ ಬಾರಿಗೆ ಬುಡಕಟ್ಟು ಜನಾಂಗೇತರ ಸಿಎಂ ಆಗಿ ರಘುಬರ್ದಾಸ್ ಅಧಿಕಾರವಹಿಸಿಕೊಳ್ಳಲಿದ್ದಾರೆ. ಈ ಹಿಂದಿನ ಸರ್ಕಾರದಲ್ಲಿ ೨೦೦೯-೧೦ ನೇ ಸಾಲಿನಲ್ಲಿ ದಾಸ್ ಉಪಮುಖ್ಯಮಂತ್ರಿಯಾಗಿ ಅಧಿಕಾರ ಚಲಾಯಿಸಿದ ಅನುಭವ ಹೊಂದಿದ್ದಾರೆ. ಅಲ್ಲದೆ ಈ ಹಿಂದೆ ಬಿಜೆಪಿಯ ರಾಷ್ಟ್ರೀಯ ಉಪಾಧ್ಯಕ್ಷರಾಗಿಯೂ ಕೂಡ ಸೇವೆಸಲ್ಲಿಸಿದ್ದರು.

- **Transliteration of above Kannada Text Example:**

jaarKhaNDana nootana muuKhyamaNtri abhyartiyaagi maaji Diciem raghubar daas aaykeyannu iMdu illi naDedas`aasakaMga sabheyalli aMtimagoLisalaagide eMdu bijepi muulagaLu tiLisive. illi naDeda bijepi s`aasakaMga pakshada sabheyalli daas avarannu aMtimavaagi pakshada ellaa s`aasakara sammatiyoMdige mukhyamaNtri abhyartiyaagi ghooshisalaayitu eMdu tiLidu baMdiddu, ii hiMde keevala buDakaTTu janaaNgada mukhyamaNtri sthankkeeruttidda. jarkhaNDnalli idee modal baarige buDakaTTu janaaNgeetara siem aagi raghubar daas adhikaara vahisikoLLliddaare. ii hiNdina sarkaradalli 2009-10 saalinalli daas upa mukhyamaNtriyaagi adhikaara chalaayisida anubhava hoNdiddaare. allade ii hiNde bijepiya raashTriya upaadyaksharaagiyuu kuuDa seeve sallisiddaru.

Sample of training corpus is shown in table 7.5.

Table 7.5. Resultant Training Corpus

jarkhaNDna	NEL	ಜಾರ್ಖಂಡ್‌ನ	naDeda	OTHER	ನಡೆದ
nootana	OTHER	ನೂತನ	S`aasakaaNga	NEO	ಶಾಸಕಾಂಗ
muuKhyamaNtri	NED	ಮುಖ್ಯಮಂತ್ರಿ	Sabheyalli	OTHER	ಸಭೆಯಲ್ಲಿ
Abhyartiyaagi	OTHER	ಅಭ್ಯರ್ಥಿಯನ್ನಾಗಿ	aMtimagoLisalaagide	OTHER	ಅಂತಿಮಗೊಳಿ ಸಲಾಗಿದೆ
Maaji	NED	ಮಾಜಿ	eMdu	OTHER	ಎಂದು
Diciem	NETE	ಡಿಸಿಎಂ	Bijepi	NEO	ಬಿಜೆಪಿ
Raghubar	NEP	ರಘುಬರ್	muulagaLu	OTHER	ಮೂಲಗಳು
Daas	NEP	ದಾಸ್	tiLisive	OTHER	ತಿಳಿಸಿವೆ
Aaykeyannu	OTHER	ಆಯ್ಕೆಯನ್ನು	.	OTHER	.
iMdu	NET	ಇಂದು			
Illi	OTHER	ಇಲ್ಲಿ			

7.7.2 Sample Test Data in Transliterated Version

navadehali: tamma sarkaarada meele baMdiruva bhrashTaacaara aaroopavannu soomavaara kaaMgres kaaryakaariNiyalli taLLihaakiruva pradhaani manamoohan siMg, saarvajanika ksheetradalliruva bhrashTaacaara toDeduhaaki, paaradars`aka- javaabdaariyuta aaDaLita niDalu baddha eMdu tiLisidaru. kelavaru hataas`a bhaavaneyiMda sarkaarada meele suLLu aaroopa maaDuttiddaare` eMdu aNNaa taMDada viruddha aaroopa maaDida pradhaani, hora dees`agaLalliruva kappu haNavannu ooMdee salakke vaapas taruvaMte oottaaya maaDuttiruva baabaa raamadeev viruddhavuu harihaaydaru. `kappu haNada baahugaLu ella ksheetragaLiguu haraDikoMDive eMbudannu avaru arthamaaDikoLLabeeku` eMdu cuccidaru. sarkaarada viruddha kelavaru hataas`a bhaavaneyiMda illasallada apapracaara maaDuttiddaare. sarkaaravannu viroodhisuva eekaika uddees`adiMda ii riitiya aaroopa maaDuttiddaare.

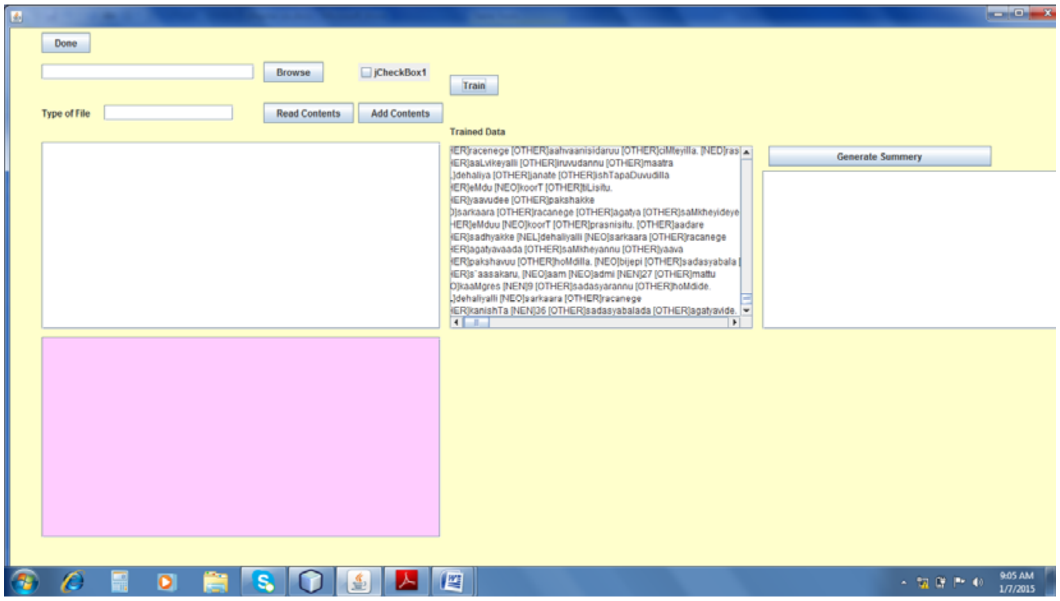
7.7.3 Sample of Tagged Test Data

navadehali/NEL:/OTHERtamma/OTHERsarkaarada/NEOmeele/OTHER baMdiruva/OTHER bhrashTaacaara/OTHERaaroopavannu/OTHER soomavaara/NET kaaMgres/NEO kaaryakaariNiyalli/OTHER taLLihaakiruva/OTHER pradhaani/NED manamoohan/NEPsiMg/NEP/OTHER, saarvajanika/OTHER ksheetradalliruva/OTHERbhrashTaacaara/OTHER toDeduhaaki/OTHER/OTHER, paaradars`aka/OTHER javaabdaariyuta/OTHER aaDaLita/OTHER niDalu/OTHER baddha/OTHER eMdu/OTHER tiLisidaru/OTHER/OTHER. Kelavaru/OTHER hataas`a/OTHER bhaavaneyiMda/OTHER sarkaarada/NEO meele/OTHER suLLu/OTHER aaroopa/OTHER maaDuttiddaare/OTHER eMdu/OTHER aNNaa/NEP taMDada/OTHER viruddha/OTHER aaroopa/OTHER maaDida/OTHER pradhaani/NED/OTHER, hora/OTHER dees`agaLalliruva/OTHER kappu/OTHER haNavannu/OTHER ooMdee/OTHER salakke/OTHER vaapas/OTHER taruvaMte/OTHER oottaaya/OTHER maaDuttiruva/OTHER baabaa/NEP raamadeev/NEP viruddhavuu/OTHER harihaaydaru/OTHER/OTHER. kappu/OTHER haNada/OTHER baahugaLu/OTHER ella/OTHER ksheetragaLiguu/OTHER haraDikoMDive/OTHER eMbudannu/OTHER avaru/OTHER arthamaaDikoLLabeeku/OTHER eMdu/OTHER cuccidaru/oTHER.

Table 7.6. Result of Hidden Markov Model

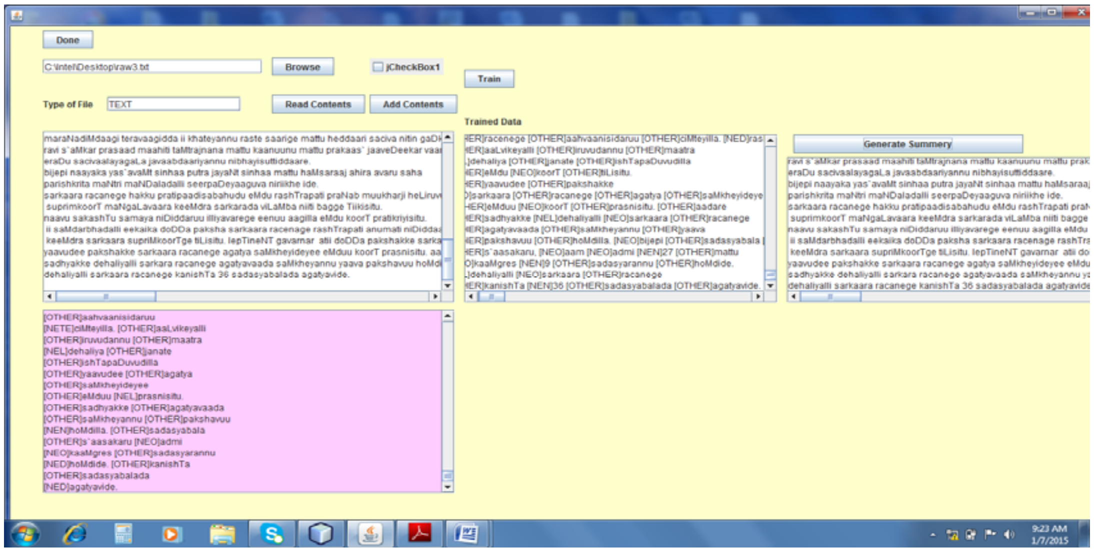
Types of NES	No. of NEs Identified
NEP	40
NEL	18
NEO	19
OTHER	-
NET	27
NEN	6
NED	24
NETE	9

Sample screen shots of HMM intermediate results and front end and final output are shown in screen shots below 7.4,7.5,7.6. and State Transition Output produced by HMM are shown in table 7.7.



```
IER]racenege [OTHER]aahvaanisidaruu [OTHER]ciMteyilla. [NED]ras
IER]aaLvikeyalli [OTHER]iruvudannu [OTHER]maatra
.]dehaliya [OTHER]janate [OTHER]ishTapaDuvudilla
IER]eMdu [NEO]koorT [OTHER]tiLisitu.
IER]yaavudee [OTHER]pakshakke
)]sarkaara [OTHER]racanege [OTHER]agatya [OTHER]saMkheyideye
IER]eMduu [NEO]koorT [OTHER]prasnisitu. [OTHER]aadare
IER]sadhyakke [NEL]dehaliyalli [NEO]sarkaara [OTHER]racanege
IER]agatyavaada [OTHER]saMkheyannu [OTHER]yaava
IER]pakshavuu [OTHER]hoMdilla. [NEO]bijepi [OTHER]sadasyabala [
IER]s`aasakaru, [NEO]aam [NEO]admi [NEN]27 [OTHER]mattu
O]kaaMgres [NEN]9 [OTHER]sadasyarannu [OTHER]hoMdide.
.]dehaliyalli [NEO]sarkaara [OTHER]racanege
IER]kanishTa [NEN]36 [OTHER]sadasyabalada [OTHER]agatyavide.
```

Figure 7.4 Screen Shot 1. Intermediate Train level of HMM Based NER



```
C:\intel\Desktop\raw3.txt
```

Type of File: TEXT

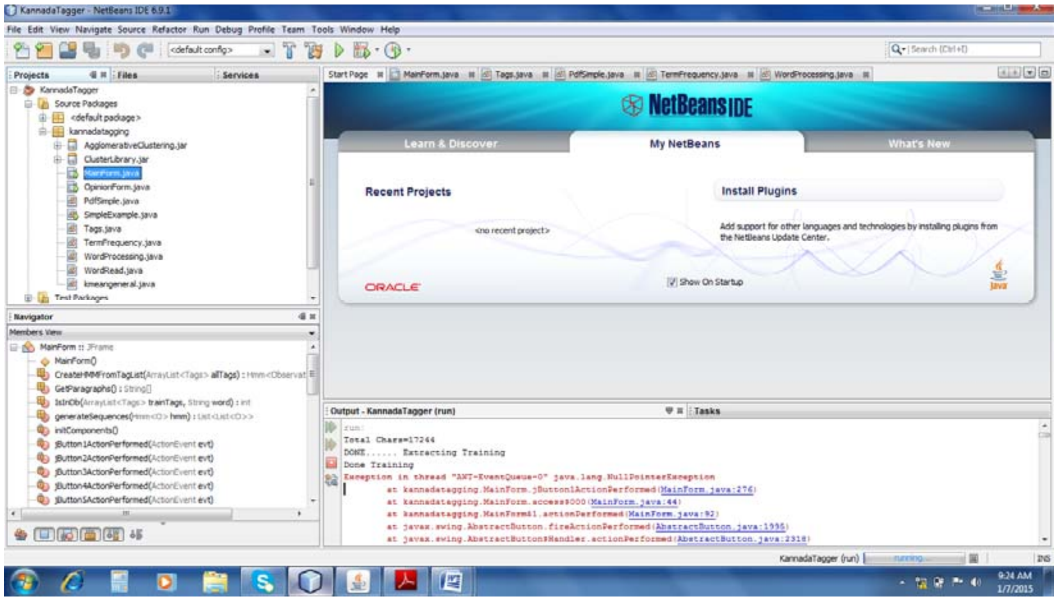
```
maraNadiMdaagi teravaagidda ii khateyannu raste saarige mattu heddaari saciva nitin gaDk
ravi s`aMkar prasaad maahiti taMtrajnana mattu kaanuunu mattu prakaas` jaaveDeekar vaar
eraDu sacivaalayagaLa javaabdaariyannu nibhayisuttiddaare.
bijepi naayaka yas`avaMt sinhaa putra jayaNt sinhaa mattu haMsaraaj ahira avaru saha
parishkrita maNtri maNDaladalli seerpaDeyaaguva niriikhe ide.
sarkaara racanege hakku pratipaadisabahudu eMdu rashTrapati praNab muukharji heLiruv
 suprimkoorT maNgaLavaara keeMdra sarkarada viLaMba niiti bagge Tiikisitu.
naavu sakashTu samaya niDiddaruu illiyavarege eenuu aagilla eMdu koorT pratikriyisitu.
 ii saMdarbhadalli eekaika doDDa paksha sarkaara racenage rashTrapati anumati niDidda
 keeMdra sarkaara supriMkoorTge tiLisitu. lepTineNT gavarnar  atii doDDa pakshakke sarka
yaavudee pakshakke sarkaara racanege agatya saMkheyideyee eMduu koorT prasnisitu. aa
sadhyakke dehaliyalli sarkara racanege agatyavaada saMkheyannu yaava pakshavuu hoMdi
dehaliyalli sarkaara racanege kanishTa 36 sadasyabalada agatyavide.
```

```
IER]racenege [OTHER]aahvaanisidaruu [OTHER]ciMteyilla. [NED]ras
IER]aaLvikeyalli [OTHER]iruvudannu [OTHER]maatra
.]dehaliya [OTHER]janate [OTHER]ishTapaDuvudilla
IER]eMdu [NEO]koorT [OTHER]tiLisitu.
IER]yaavudee [OTHER]pakshakke
)]sarkaara [OTHER]racanege [OTHER]agatya [OTHER]saMkheyideye
IER]eMduu [NEO]koorT [OTHER]prasnisitu. [OTHER]aadare
IER]sadhyakke [NEL]dehaliyalli [NEO]sarkaara [OTHER]racanege
IER]agatyavaada [OTHER]saMkheyannu [OTHER]yaava
IER]pakshavuu [OTHER]hoMdilla. [NEO]bijepi [OTHER]sadasyabala [
IER]s`aasakaru, [NEO]aam [NEO]admi [NEN]27 [OTHER]mattu
O]kaaMgres [NEN]9 [OTHER]sadasyarannu [OTHER]hoMdide.
.]dehaliyalli [NEO]sarkaara [OTHER]racanege
IER]kanishTa [NEN]36 [OTHER]sadasyabalada [OTHER]agatyavide.
```

```
ravi s`aMkar prasaad maahiti taMtrajnana mattu kaanuunu mattu prak
eraDu sacivaalayagaLa javaabdaariyannu nibhayisuttiddaare.
bijepi naayaka yas`avaMt sinhaa putra jayaNt sinhaa mattu haMsaraaj
parishkrita maNtri maNDaladalli seerpaDeyaaguva niriikhe ide.
sarkaara racanege hakku pratipaadisabahudu eMdu rashTrapati praN
 suprimkoorT maNgaLavaara keeMdra sarkarada viLaMba niiti bagge
naavu sakashTu samaya niDiddaruu illiyavarege eenuu aagilla eMdu
 ii saMdarbhadalli eekaika doDDa paksha sarkaara racenage rashTra
 keeMdra sarkaara supriMkoorTge tiLisitu. lepTineNT gavarnar  atii do
yaavudee pakshakke sarkaara racanege agatya saMkheyideyee eMdu
sadhyakke dehaliyalli sarkara racanege agatyavaada saMkheyannu ya
dehaliyalli sarkaara racanege kanishTa 36 sadasyabalada agatyavide
```

```
[OTHER]aahvaanisidaruu
[NETE]ciMteyilla. [OTHER]aaLvikeyalli
[OTHER]iruvudannu [OTHER]maatra
[NEL]dehaliya [OTHER]janate
[OTHER]ishTapaDuvudilla
[OTHER]yaavudee [OTHER]agatya
[OTHER]saMkheyideyee
[OTHER]eMduu [NEL]prasnisitu.
[OTHER]sadhyakke [OTHER]agatyavaada
[OTHER]saMkheyannu [OTHER]pakshavuu
[NEN]hoMdilla. [OTHER]sadasyabala
[OTHER]s`aasakaru [NEO]admi
[NEO]kaaMgres [OTHER]sadasyarannu
[NED]hoMdide. [OTHER]kanishTa
[OTHER]sadasyabalada
[NED]agatyavide.
```

Figure 7.5. Screen Shot : Final Output of HMM Based NER



```
run:
Total Chars=17244
DONE...... Extracting Training
Done Training
Exception in thread "AWT-EventQueue-0" java.lang.NullPointerException
        at kannadatagging.MainForm.jButton1ActionPerformed(MainForm.java:276)
        at kannadatagging.MainForm.access$000(MainForm.java:44)
        at kannadatagging.MainForm$1.actionPerformed(MainForm.java:92)
        at javax.swing.AbstractButton.fireActionPerformed(AbstractButton.java:1995)
        at javax.swing.AbstractButton$Handler.actionPerformed(AbstractButton.java:2318)
```

Figure 7.6. Screen Shot. Front End for HMM Based NER

Table 7.7. State Transition Output Produced By HMM

```
digraph G {
0 -> 0 [label=0.12];
0 -> 1 [label=0.12];
0 -> 2 [label=0.12];
0 -> 3 [label=0.12];
0 -> 4 [label=0.12];
0 -> 5 [label=0.12];
0 -> 6 [label=0.12];
0 -> 7 [label=0.12];
1 -> 0 [label=0.12];
1 -> 1 [label=0.12];
1 -> 2 [label=0.12];
1 -> 3 [label=0.12];
1 -> 4 [label=0.12];
1 -> 5 [label=0.12];
1 -> 6 [label=0.12];
1 -> 7 [label=0.12];
2 -> 0 [label=0.12];
```

```
2 -> 1 [label=0.12];
2 -> 2 [label=0.12];
2 -> 3 [label=0.12];
2 -> 4 [label=0.12];
2 -> 5 [label=0.12];
2 -> 6 [label=0.12];
2 -> 7 [label=0.12];
3 -> 0 [label=0.12];
3 -> 1 [label=0.12];
3 -> 2 [label=0.12];
3 -> 3 [label=0.12];
3 -> 4 [label=0.12];
3 -> 5 [label=0.12];
3    -> 6 [label=0.12];
3 -> 7 [label=0.12];
4 -> 0 [label=0.12];
4 -> 1 [label=0.12];
4 -> 2 [label=0.12];
4 -> 3 [label=0.12];
4 -> 4 [label=0.12];
4 -> 5 [label=0.12];
4 -> 6 [label=0.12];
4 -> 7 [label=0.12];
5 -> 0 [label=0.12];
5 -> 1 [label=0.12];
5 -> 2 [label=0.12];
5 -> 3 [label=0.12];
5 -> 4 [label=0.12];
5 -> 5 [label=0.12];
5 -> 6 [label=0.12];
5 -> 7 [label=0.12];
6 -> 0 [label=0.12];
6 -> 1 [label=0.12];
6 -> 2 [label=0.12];
```

```
6 -> 3 [label=0.12];
6 -> 4 [label=0.12];
6 -> 5 [label=0.12];
6 -> 6 [label=0.12];
6 -> 7 [label=0.12];
7 -> 0 [label=0.12];
7 -> 1 [label=0.12];
7 -> 2 [label=0.12];
7 -> 3 [label=0.12];
7 -> 4 [label=0.12];
7 -> 5 [label=0.12];
7 -> 6 [label=0.12];
7 -> 7 [label=0.12];
0 [shape=double circle, label="0 - Pi= 0.04 - [ Discrete distribution --- NEL 0.041,
NET 0.044, NEP 0.106, NEO 0.057, OTHER 0.644, NED 0.077, NETE 0.015, NEN
0.015 ]"];
1 [shape=double circle, label="1 - Pi= 0.04 - [ Discrete distribution --- NEL 0.125,
NET 0.125, NEP 0.125, NEO 0.125, OTHER 0.125, NED 0.125, NETE 0.125, NEN
0.125 ]"];
2 [shape=double circle, label="2 - Pi= 0.11 - [ Discrete distribution --- NEL 0.125,
NET 0.125, NEP 0.125, NEO 0.125, OTHER 0.125, NED 0.125, NETE 0.125, NEN
0.125 ]"];
3 [shape=double circle, label="3 - Pi= 0.06 - [ Discrete distribution --- NEL 0.125,
NET 0.125, NEP 0.125, NEO 0.125, OTHER 0.125, NED 0.125, NETE 0.125, NEN
0.125 ]"];
4 [shape=double circle, label="4 - Pi= 0.64 - [ Discrete distribution --- NEL 0.125,
NET 0.125, NEP 0.125, NEO 0.125, OTHER 0.125, NED 0.125, NETE 0.125, NEN
0.125 ]"];
5 [shape=double circle, label="5 - Pi= 0.08 - [ Discrete distribution --- NEL 0.125,
NET 0.125, NEP 0.125, NEO 0.125, OTHER 0.125, NED 0.125, NETE 0.125, NEN
0.125 ]"];
6 [shape=double circle, label="6 - Pi= 0.02 - [ Discrete distribution --- NEL 0.125,
NET 0.125, NEP 0.125, NEO 0.125, OTHER 0.125, NED 0.125, NETE 0.125, NEN
0.125 ]"];
```

7 [shape=double circle, label="7 - Pi= 0.02 - [Discrete distribution --- NEL 0.125, NET 0.125, NEP 0.125, NEO 0.125, OTHER 0.125, NED 0.125, NETE 0.125, NEN 0.125]"];

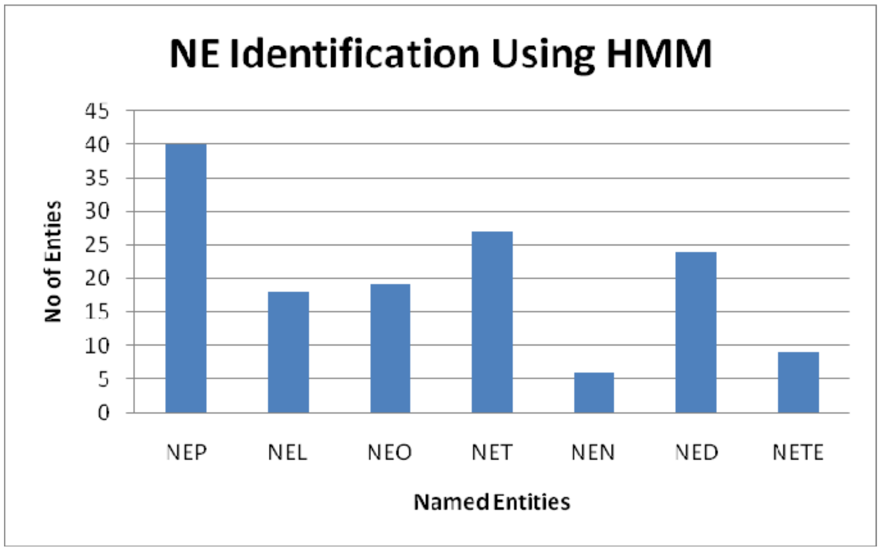


Figure 7.7. Graph of HMM Based NE Identification

7.8. Performance Matrix

Performance matrix mathematically defines the way to measure a system's performance against the human annotated corpus. The system's performance is measured in terms of Precision (P), Recall I, and F-Measure (F).

- *Precision (P):* Precision is the fraction of the documents retrieved that are relevant to the user's information need.

Precision (p) = (correct-answer) / (answer-produced)……………………….. (7.20)

=143/167= 85%

- *Recall I:* Recall is the fraction of the documents that are relevant to the query that are successfully retrieved.

Recall I: (correct-answer) / (total possible correct-answers)………………….. (7.21)

143/150=95%

- *F-Measure:* The weighted harmonic mean of precision and recall, the traditional F-measure or balanced F-score is

F-Measure (F): (β2+1) PR/ (β2R+P). β2 is the weighting between precision and recall typically β=1. When recall and precision are evenly weighted, i.e. β=1, F-measure is called F1 measure.

F1-measure=2*P*R/ (P+R) ……………………………………………….. (7.22)

=2 *85.6*95.33/(85.6+95.33)=93%

There is a tradeoff between precision and recall in the performance metric.

• Summary

HMM based algorithm is developed for Kannada NER. and tested using various test cases. Viterbi algorithm computes the optimal state sequence for the given test sentence.

The state transitions are depicted pictorially by using an **additional software graphviz which draws the state transition diagram.** The HMM is designed to take **input from pdf document as well apart from input from txt file or doc and docx files that is the speciality here**. Performance can be improved by increasing the training data. HMM is a generative model and it gives the output directly by modeling the transition matrix based on the training data. The results can be improved by providing more data points, but there is no direct control over the output labels. HMM learns the transition probabilities on its own based on the training data provided. Hence if we provide more data points, then we can improve the model to include a wider variety.

Chapter 8

Named Entity Recognition and Classification using Conditional Random Field

In previous chapters, A Hidden Markov Model (HMM) machine learning approach for NER is discussed. In this chapter we are using Conditional Random Field (CRF) machine learning technique to recognize and classify named entities in Kannada with long term dependencies. Conditional Random Fields (CRFs) are a probabilistic framework for labeling sequential data based on the conditional approach. **The primary advantage of the CRF over the HMM is the conditional nature and relaxation of the independence assumption**. We have achieved higher accuracy in CRF approach than the in HMM approach. The accuracy of classification is more accurate in CRF approach due to flexibility of adding more features unlike joint probability alone as in HMM. In HMM it is not practical to represent multiple overlapping features and long term dependencies. CRF ++ Tool Kit is used for experimentation.

8.1 Introduction to Conditional Random Field

CRF is natural combination of diverse set of features which is a very flexible method and features are easily incorporated unlike HMM. CRF is probabilistic framework for labeling and segmenting structureing the data in the form of sequences, trees and lattices. CRF is especially useful in modeling time-series data where the temporal dependency can manifest itself in various different forms. The underlying idea is that of defining a conditional probability distribution over label sequences, given a particular observation sequence, rather than a joint distribution over both label and observation sequences. The primary advantage of CRF is the relaxation of the independence assumption which says that the variables do not depend on each other and they don’t affect each other in any way.

8.2 Hidden Markov Model versus Conditional Random Field

One of the most common methods of performing POS sequence labeling task is that of employing Hidden Markov Models (HMMs) to identify the most likely POS tag sequence for the words in a given sentence. HMMs are generative models, which maximize the joint probability distribution *p(X, Y)* where *X* and *Y* are random variables respectively, representing the observation sequence (i.e. the word sequence in a sentence) and the corresponding label sequence (i.e. the POS tag sequence for the word of a sentence). Due to the joint probability distribution of the generative models. This assumption may work for a simple data set. However for the problem of the POS labeling task, **the observation sequence may depend on multiple interacting features and long distance dependencies. One way to satisfy the above criteria is to use a model that defines conditional probability P *(Y|X)* over label sequences given a particular observation sequence (X), rather than a joint probability distribution over both label and observation sequence**. Conditional models are used to label an unknown observation sequence, by selecting the label sequence that maximizes the conditional probability.

CRF prevents the *label bias* problem of the Maximum Entropy (ME) model and has an advantage over other directed graphical models. (Lafferty et. al., 2001),(Pinto and McCallum, 2003),(Sha and Pereira, 2003) propose that CRF better-performs HMM and ME models on large number of sequence labeling tasks. A CRF parameter is globally conditioned on *X*, the random variable representing the observation sequence is formally a graph *G= (V, E),* which is a undirected graph such that, there is a node $v \in V$ corresponding to each of the random variables representing an element Y_v of *Y where Y is POS tag label*. If each random variable Y_v follows the Markov chain property with respect to *G*, then *(Y, X)* is a conditional random field. However, when we model the POS sequence labeling problem, the simplest and the most common graph structure encountered is that in which the nodes corresponding to elements of *Y* i.e. the POS tag labels form a simple first order chain as illustrated in figure 8.1.

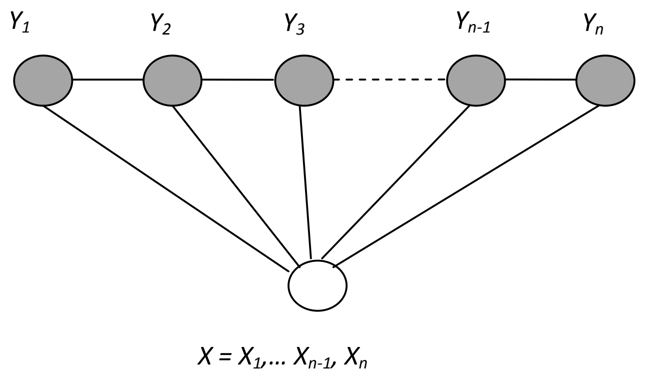


Figure 8.1. Graphical Structure of a Chain-Structured CRF for sequences.

8.3 Statistical Background

As per Lafferty et al. y is the probability of label sequence given the observation sequence *x* being a normalized product of the potential functions, each of the form

$$\exp\left(\sum_j \lambda_j t_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k s_k(y_i, x, i)\right) \qquad \text{...(8.1)}$$

Where $t_j(y_{i-1}, y_i, x, i)$ is a transition feature function of the entire observation sequence and the labels at position *i* and *i-1* are the label sequence. The function $s_k(y_i, x, i)$ is a state feature function of the label at position *i* and the observation sequence λ_j and μ_k are the model parameters to be estimated from the training data, while defining feature functions in CRF model, we construct a set of real valued features $b(x,\ i)$ of the observation to express some characteristics of the training data. Each feature function takes the real valued observation features $b(x,\ i)$. The current state, or current and previous states takes some particular values. Thus all feature functions are real valued in nature. For example, consider the following feature functions.

$$t_j(y_{i-1}, y_i, x, i) = \begin{cases} b(x,i) & \text{if } y_{i-1} = NN \text{ and } y_i = PP \\ 0 & \text{otherwise} \end{cases}$$ ..(8.2)

This allows the probability of a label sequence y given an observation sequence x to be written as

$$F_j(y,x) = \sum_{j=1}^{n} f_j(y_{i-1}, y_i, x, i)$$ ... (8.3)

where $f_j(y_{i-1}, y_i, x, i)$ is either a state function $s_k(y_i, x, i)$ or a transition function $t_j(y_{i-1}, y_i, x, i)$, Z(x) is the normalizing factor.

8.4 Parameter Estimation

Assuming the training data $\{(x^{(k)}, y^{(k)})\}$ are independen and identically distributed, the product of equation 8.3 over all training sequences, as a function of the parameter λ, is known as the likelihood, denoted by $p(\{y^{(k)}\} \mid \{x^{(k)}\}, \lambda)$. The parameter in Maximum likelihood training is chosen in such way that the logarithm of the likelihood, is known as the log-likelihood, is maximized for a CRF. The log-likelihood is given by

$$L(\lambda) = \sum_k \left[\log \frac{1}{Z(x^{(k)})} + \sum_j \lambda_j F_j(y^{(k)}, x^{(k)}) \right]$$ ...(8.4)

The function is concave, guarantees to convergence to the global maxima. Differentiating the log-likelihood with respect to the parameter λ_j given as

$$\frac{\partial L(\lambda)}{\partial \lambda_j} = E_{\tilde{p}(Y,X)}\left[F_j(Y,X)\right] - \sum_k E_{p(Y|x^{(k)},\lambda)}\left[F_j(Y,x^{(k)})\right]$$ ... (8.5)

Where $\tilde{p}(Y,X)$ is the empirical distribution of training data and $E_p[]$ denotes the expectation with respect to distribution of p. The feature expected value with respect to the model distribution is equal to the expected value under the empirical distribution of the training data. It is not possible to analytically estimate the parameter values that maximize the log-likelihood. Setting the gradient to zero and solving for λ does not always give a closed form of solution. Instead, maximum likelihood parameter must be estimated, using an iterative technique such as iterative scaling (Darroch and Ratcliff, 1972), (Berger, 1997), (Pietra et. al., 1995) or the gradient based method (Sha and Pereira, 2003), (Wallach, 2002).

CRF permits to statistically capture the characteristics of a native language. It is a finite state model with un-normalised transition probability. CRF also generalizes easily to analogues of stochastic context-free grammar that would be useful in problems like Natural Language Processing. Conditional model which is used to label an observation sequence $0 = (o_1, o2, OT)$ by selecting the label sequence S= (s_1, s_2, .s_T) which maximizes the conditional probability P $(S|O)$.

$$P\ (S/O)\ \alpha\ exp\ (\textstyle\sum_{t=1}^{T} \sum_k \lambda_k \times f_k\ (s_{t-1},\ s_t,\ o,\ t) \qquad \textbf{... (8.6)}$$

Where $f_k\ (st_{-1},\ s_t,\ o,\ t)$ is a feature function, whose weight is learned via training. The value of a feature function is either 0 or 1 and k is number of features.

$$f_k\ (st_{-1},\ s_t,\ o,\ t) = \begin{cases} b(o,\ t)\ if\ s_{t-1} = B\text{-}NEP\ and\ s_t = I\text{-}NEP \\ 0 \quad otherwise \end{cases} \qquad \textbf{... (8.7)}$$

$$\mathrm{b\ (o,\ t)} = 1 \quad \begin{cases} \text{if the observation sequence at position i = "NER"} \\ 0, \quad \text{otherwise.} \end{cases}$$

In the next section the proposed architecture for named entity recognition using conditional random field is described.

8.5 Proposed NER System Using Conditional Random Field

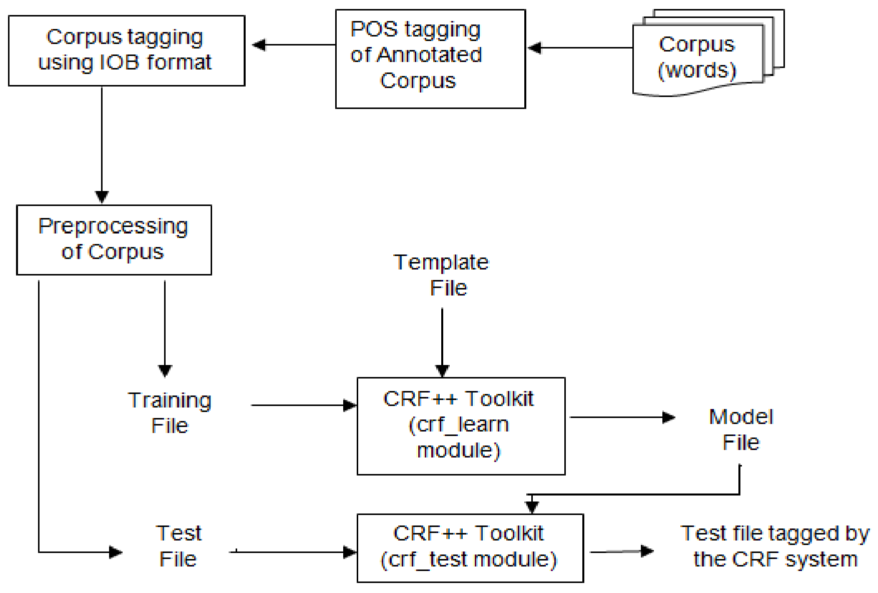

Corpus tagging using IOB format
POS tagging of Annotated Corpus
Corpus (words)
Preprocessing of Corpus
Template File
Training File
CRF++ Toolkit (crf_learn module)
Model File
Test File
CRF++ Toolkit (crf_test module)
Test file tagged by the CRF system

Figure 8.2. Proposed NER System Using CRF

8.5.1 Corpus Tagging Scheme Using IOB Format

The corpus is tagged using the IOB (Immediately outside Beginning) tagging scheme, shown in table 8.1.

- Empty lines represent sentence boundaries.
- Whenever two entities of type XXX are immediately next to each other, the first word of the second entity will be tagged B-XXX in order to show that it starts another entity. The tag I-XXX is used for words inside a named entity of type XXX.
- Words tagged with O are outside of named entities.
- Separate columns are attached for each list.
- Length feature is also added during this stage.

- ➢ After applying NE and POS tagging, the corpus is further preprocessed for adding Prefixes, Suffixes for each word.
- ➢ Gazetteer lists are also added here. If the word is present in any of the lists, then entry corresponding to that list is set to 1, otherwise it is 0.

Table 8.1. Shows IOB Tagging Scheme Used in CRF

Transliterated Tokens	POS	IOB
Narendra	NNP	B-NEP
Moodi	NNP	I-NEP
Pradhaani	NN	NED
aadaru	VBZ	O

8.5.2 Preparing Training File and Test File Formats Using IOB Format

CRF++ toolkit demand the data files to be in a particular format of multiple columns separated by single white space. Here the features are considered in 28 columns as shown in below box 8.1, and also in 3 column format. The first column contains the current word, second word contains POS feature in one column, the third column is IOB format which contains one column, the fourth column contains the word length and next column contains the prefixes and suffixes. Prefix and suffix window contains window size 5 to 7. The next is gazetteer list, which contains 12 columns. The last column is the true answer tag. The following are the feature samples.

- Context Word :
- Parts of Speech (POS) Information
- Word Prefix: we have experimented with a Word prefix, of length 1 to 4 characters, of the current word as a feature.

- Word Suffix: we have experimented with a Word suffix, of length 1 to 7 characters, of the current word as a feature.
- Named Entity Information: NE tag of the previous word and next word is used as a feature.
- Gazetteer Lists: We have developed 12 gazetteer lists.

  -These lists have been used as the binary valued features of the CRF.

  - If the current token is in a particular list then the corresponding feature is set to 1 for the current and/or the surrounding word(s), otherwise, set to 0.

Gazetteer list contain Location name, First name, Middle name, Last name, Organization name, Person Prefix, Day and Month lists, Abbreviation list etc. Consider the sentence below to show the sample format of training file.

Kannada Sentence:

ಯುವರಾಜಸಿಂಗನನ್ನು ಎಷ್ಯಾ ಕಪ್ ಟೀಮ್ ಇಂಡಿಯಾದಲ್ಲಿ ಸೇರಿಸಿಕೊಳ್ಳಲಾಗಿದೆ.

Transliteration:

yuuvaraaj siMganannu eshyaa kap Tiimnalli seerisikoLLalagide.

English:

Yuvaraaj singh is included in Asia Cup.

```
ಯುವರಾಜNNP B-NEP MORE ವರಾಜ ರಾಜಜLL LLಯುವರಾಜಯುವರಾಯುವಯು LL

LL1 0 0 0 0 0 0 0 0 0 0 0 B-NEP
ಸಿಂಗನನ್ನುNNP I-NEP MORE ಂಗನನ್ನುಗನನ್ನು ನನ್ನು ನ್ನು್ನುLL

ಸಿಂಗನನ್ನುಸಿಂಗನಸಿಂಗ ಸಿಂ ಸಿ0 0 0 0 0 0 0 0 0 0 0 0 I-NEP

ಏಷ್ಯಾ NN B-NETE MORE ಏಷ್ಯಾ ಷ್ಯಾ ್ಯಾLL LL LL LL LLಏಷ್ಯಾ ಏಷಾ ಎ0 0 0 1 0 0

0 0 0 0 0 0 B-NETE
ಕಪ್NN I-NETE LESS LL LL ಕಪ್ಪ್ ಫLL LL LL ಕಪ್ಪ್ಕ 0 0 0 1 0 0 0 0 0 0

0 0 I-NETE
ಟೀಮ್NN I-NETE LESS LL ಟೀಮ್ಮ್ಮLL LL LL LL ಟೀಮ್ಟೀಮಟಿ0 0 0 1 0 0

0 0 0 0 0 0 I-NETE
ಇಂಡಿಯಾದಲ್ಲಿNN I-NETE MORE LLಂಡಿಯಾದಲ್ಲಿ ಡಿಯಾದಲ್ಲಿ ಯಾದಲ್ಲಿ

ದಲ್ಲಿ ಲ್ಲಿLLಇಂಡಿಯಾದ ಇಂಡಿಯಾ ಇಂಡಿ ಇಂ000 100000000I-NETE

ಸೇರಿಸಿಕೊಳ್ಳಲಾಗಿದೆVBZMORE LL ರಿಸಿಕೊಳ್ಳಲಾಗಿದೆ ಕೊಳ್ಳಲಾಗಿದೆ ಳ್ಳಲಾಗಿದೆ ಲಾಗಿದೆ

ಗಿದೆ ದೆLLಸೇರಿಸಿಕೊಳ್ಳಲಾ ಸೇರಿಸಿಕೊಳ್ಳ ಸೇರಿಸಿಕೊ ಸೇರಿಸಿ ಸೇರಿ ಸೇಂ 0 0 0 0 0 0 0 0 0 0
```

Box 8.1. Format of Training File and Testing File for Conditional Random Field

8.5.3 Preparing Feature Templates

Template file is changed in accordance with the features that we want to consider for training and test files. The information considered here are as follows.

- A template file is prepared, in which all the columns except the last column of data sets are used as feature templates, to be used by CRF++ toolkit.
- Each line in the template file denotes one template.
- In each template, %x [row, col] is used to specify a token in the input data, row specifies the relative position from the current focusing token and column.This Specifies the absolute position of the column. Here's is an example of template file shown in below table.

Table 8.2. Shows Feature Templates

U00:%x[0,0]
U01:%x[-1,0]
U02:%x[1,0]
U03:%x[-2,0]
U04:%x[2,0]
U05:%x[0,1]
U06:%x[1,1]
U07:%x[-1,2]

CRF Toolkit makes the following replacements in the template file during execution as per the token under consideration.

8.5.4 Feature (attribute) generation

The next step is to preprocess the training and testing data to extract attributes that express the characteristics of words (items) in the data. CRF suite generates internally features from attributes in a data set. In general, this is the most important process for machine-learning approaches because a feature design greatly affects the labeling accuracy. In this work the following kinds of attributes from a word at position t in offsets from the beginning of a sequence are extracted.

- w[t-2], w[t-1], w[t], w[t+1], w[t+2],
- w[t-1]|w[t], w[t]|w[t+1],
- pos[t-2], pos[t-1], pos[t], pos[t+1], pos[t+2],
- pos[t-2]|pos[t-1], pos[t-1]|pos[t], pos[t]|pos[t+1], pos[t+1]|pos[t+2],
- pos[t-2]|pos[t-1]|pos[t], pos[t-1]|pos[t]|pos[t+1], pos[t]|pos[t+1]|pos[t+2]

Here pos[t] and w[t] present the part-of-speech and word at position t in a sequence. The features considered express the characteristic of the word at position t by using information from surrounding words, e.g., w[t-1] and pos[t+1].

8.6 Results and Experiments

We considered a sample file of size 1000 words, and we observed that, the results obtained using CRF technique is more accurate as compared to HMM technique. But however the amount of time required in designing training and testing data is more in CRF technique as compared to HMM technique. Both techniques work better if there is more training data. Here training data is tagged data. We manually annotated a corpus of 2,100 words. We have used adaptations of two machine learning methods, namely Hidden Markov Model (HMM), and Conditional random Field (CRF). The number of named entities recognized is shown in table. And figure 8.3 gives the number of entity types recognized in the input file.

Table 8.3. Named Enties Identified By CRF

Named Entity Types	No of Named entities
NEP	48
NEL	14
NEO	25
NET	12
NEN	11
NED	25
NETE	3

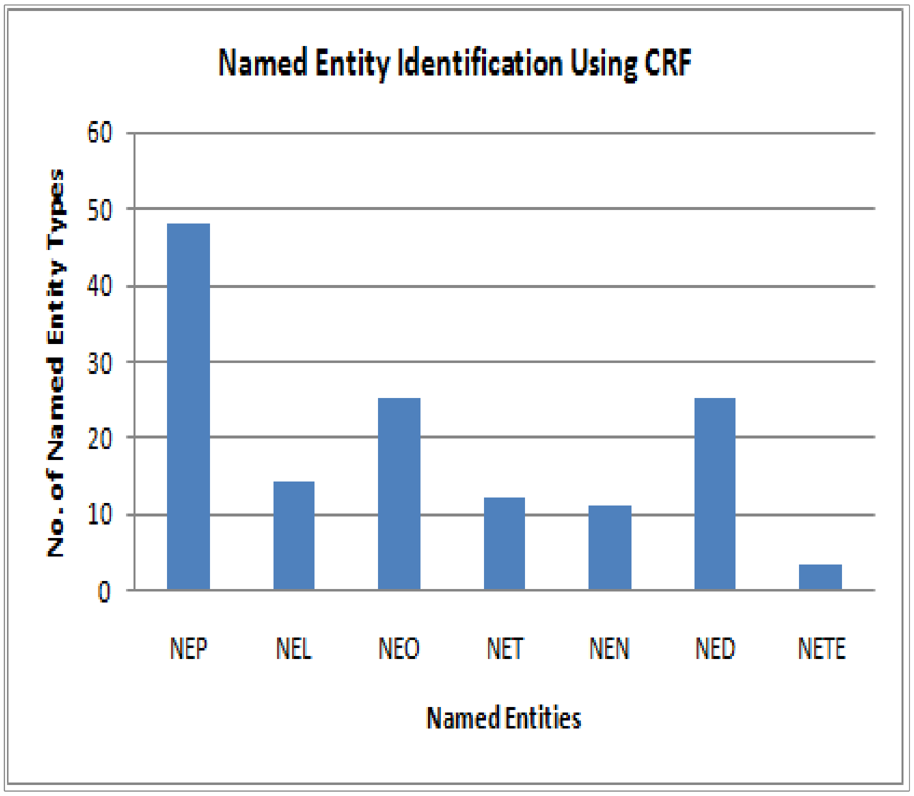
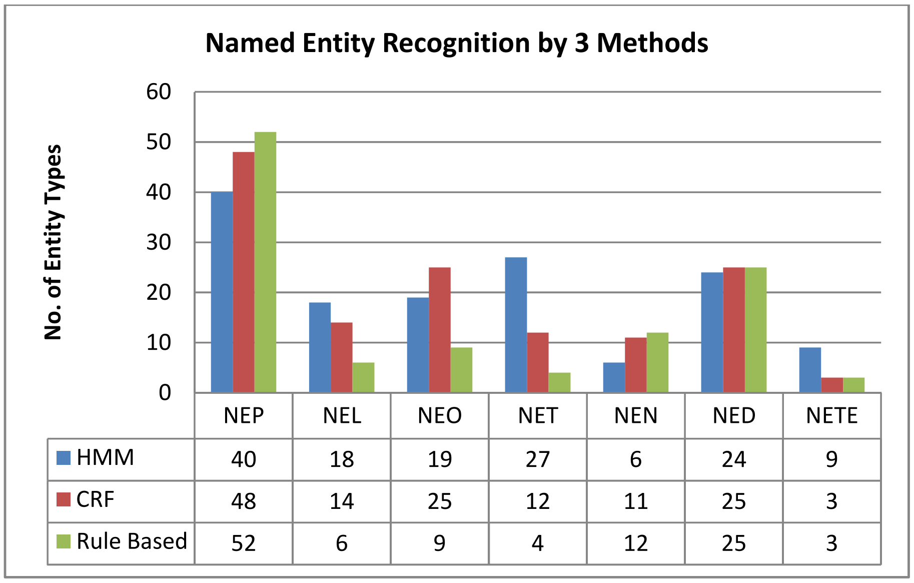


Figure 8.3. Named Entity recognition by CRF

The named entity recognition for a file of 1000 words is experimented by 3 methods is shown in figure below. Accuracy of classification is more accurate in CRF in case of identifying organization entities as CRF is capable of identifying of long entities. However the size of organization eneties varies, it is not of fixed length. Rule based method regarding identifying long named entitied needs to be modified.



	NEP	NEL	NEO	NET	NEN	NED	NETE
🟦 HMM	40	18	19	27	6	24	9
🟥 CRF	48	14	25	12	11	25	3
🟩 Rule Based	52	6	9	4	12	25	3

Figure 8.4. Named Entity Recognition by 3 Methods

- **Use *crflearn* command (Encoding):**

The following command is used for training the system.

Syntax: % `crf_learn template file train file model file`

Where *template file* and *training file* are the files which we have prepared in advance, *Crf_learn* generates the trained model file in *model file*. This model file is then used by CRF++ for tagging the test data.

- **Use of *crf_test* command (Decoding):**

Purpose: The testfile is the test data. This file has to be written in the same format as the training file.

Syntax: `% crf_test -m model file test_files`

Example: `crf_test –m model test.data`

To get the output in the file use the following command.

Purpose: The purpose is to show the output file

Example: `crf_test –m model test.data >> output.data`

Table 8.4. CRF learn Output

```
CRF++: Yet another CRF Tool Kit
Copyright (C) 2005-2008 Taku Kudo, All rights reserved.
reading training data:
Done!0.00 s
Number of sentences: 1
Number of features:  102
Number of thread(s): 1
Freq:            1
eta:            0.00010
C:              1.00000
shrinking size:      20
iter=0 terr=0.80000 serr=1.00000 act=102 obj=3.46574diff=1.00000
iter=1 terr=0.00000 serr=0.00000 act=102 obj=1.03147diff=0.70238
iter=2 terr=0.00000 serr=0.00000 act=102 obj=0.96499diff=0.06445
iter=3 terr=0.00000 serr=0.00000 act=102 obj=0.95913diff=0.00607
iter=4 terr=0.00000 serr=0.00000 act=102 obj=0.95870diff=0.00046
iter=5 terr=0.00000 serr=0.00000 act=102 obj=0.95867diff=0.00003
iter=6 terr=0.00000 serr=0.00000 act=102 obj=0.95867diff=0.00000
iter=7 terr=0.00000 serr=0.00000 act=102 obj=0.95867diff=0.00000
Done!0.05 s
```

- ***Sample Tagged output by CRF***

```
ಯುವರಾಜ NNP B-NEP MORE ವರಾಜ ರಾಜಜ 0 0 ಯುವರಾಜ ಯುವರಾ ಯುವ
ಯುLL LL1 0 0 0 0 0 0 0 0 0 0 0 B-NEPB-NEP
ಸಿಂಗನನ್ನು NNP I-NEP MORE ಂಗನನ್ನುಗನನ್ನು ನನ್ನು ನ್ನು ್ನ 0 0 ಸಿಂಗನನು
್ನಸಿಂಗನ ್ಸಿಂಗ ಸಿಂ ಸಿ೦ ೦ ೦ ೦ ೦ ೦ ೦ ೦ ೦ ೦ ೦ ೦ I-NEPI-NEP
ಏಷ್ಯಾ NN B-NETE MORE ಏಷ್ಯಾ ಷ್ಯಾ ್ಯಾLL LL LL LL LLಏಷ್ಯಾ ಏಷಾ ಏ 0 0 0 1
0 0 0 0 0 0 0 0 B-NETEB-NETE
```

ಕಫ್ NN I-NETE LESS LL LL ಕಫ್ ಫ್ ಫ LL LL LL ಕಫ್ ಕ ಫ ಕ 0 0 0 1 0

0 0 0 0 0 0 0 I-NETE*I-NETE*

ಟಿಮ್ I-NETE LESS LL ಟಿಮ್ ಮ್ ಮ LL LL LL LL ಟಿಮ್ ಟಿವ ಟಿ 0 0 0

1 0 0 0 0 0 0 0 0 I-NETE*I-NETE*

- **Summary**

CRF is a discriminative model which outputs a confidence measure. This is really useful in most cases because we want to know how sure the model is about the label at that point. This confidence measure can be the threshold to suit various applications. The good thing about the confidence measure is that the number of false alarms is low compared to HMM.

The primary advantage of CRFs over HMMs is their conditional nature, resulting in the relaxation of the independence assumptions required by HMMs. Additionally, CRF avoid the label bias problem, a weakness exhibited by Markov models based on directed graphical models. CRF outperform HMMs on a number of real-world sequence labeling tasks.

The power of the CRF model lies in its diverse and overlapping set of features. Instead, HMM uses local features (*current word, previous one or two tags*) for POS tagging. In CRF experiment more number of features (*current word, previous tag, prefix/suffix of length four*) are used. This may be one of the reasons for the relatively lesser accuracy of the tagging task in HMM. A large number of features works well when a large amount of annotated training data is available to find a significant amount of every feature instance.

However, the performance of the current CRF asunder system is not as good as that of the contemporary system available for English and other European languages. Performance can be increased by improving the training data and increasing the features considerations.

Chapter 9

Conclusion and Scope for Future Work

This chapter summarizes the work carried out here and proposes work that can be carried out in future in sequel to this work. We have defined a process to build automatic morphosyntactic annotator system for Kannada. In the process of development, the resources developed can be used further for future work on processing of Kannada language. The resources include Kannada electronic dictionary, hierarchical POS tagset, morphological analyzer/generator and named entity recognizer. The chapter has three subsections. First, a brief on work carried out in the thesis is presented. Then, the next section lists the objectives met in the thesis. Lastly, a section briefs on future work.

9.1 Work Carried Out

- **An exhaustive morphological analyzer and generator using finite state transducers are developed.** *Aspects of inflectional morphology, derivational morphology and also external saMdhi (for the first time) are handled here. The morph module uses a hierarchical tagset instead of a flat tagset.* Morphological analyzer/generator system gives complete details involved in word formation / analysis process morpheme by morpheme, and this kind of morph system can be used *as language learning tool and* these resources are useful in further NLP applications like Machine translation or parsing applications where detailed information is crucial. *The morphological analyzer/generator developed here is the first system of its kind built using a hierarchical tagset for Kannada as compared to the few existing systems that were designed using a flat tagset viz. (Prof. K N Murthy, 1999), (IIIT-Hyderabad, 2007), (Shambhavi, 2011), (Shalini urs et al., 2007).* The tagging accuracy of our morphological analyzer/generator is more than 90% for nouns, 100% for pronouns and 85% for verbs.

- **Morph related electronic dictionary of around 30,000 words is developed for the purpose of morphological analyzer/generator system**. Various techniques *like pattern matching, filtering, frequency estimation and regular* *expressions are used in developing dictionary. Knowledge of native language is crucial in building electronic dictionaries for native languages. 126 heuristic pattern rules are used in extracting words from corpus. CIIL, DoE corpus of size 3 million words is used for building the dictionary. The dictionary is built using hierarchical tag set unlike the existing dictionaries which uses flat tag set. The information related for morphological generation process is also stored in the dictionary; this information avoids our morphological generator system to be over general. The dictionary developed in this work is the first dictionary built using hierarchical tagset.* Developing an electronic dictionary using hierarchical tag is valuable effort because each word has to be marked for its gender, number, countable and uncountable features. We observe that available dictionaries do not give detailed morpho syntactic information and often do not agree with one another even with respect to the main grammatical category, this happens because the basis for categorization could be somewhat different in each case. A noun can act like an adjective and modify another noun and so one dictionary may treat it as a noun while the other may treat it as an adjective. Locative nouns are often treated as adverbs of place. We need to have a consistent and clearly defined set of major grammatical categories. Here we use morphological and syntactic considerations only. Thus any word which can take a plural can only be a noun and not an adjective. If a case marker can be added, it can only be a noun, not an adverb. We have seen that electronic lexicons are extremely useful both for human users and for NLP programs.

- **Hierarchical part of speech (HPOS) tag set is developed at the initial stage of research work. 171 atomic tag elements are used in designing the tag set. The tagset plays a vital role in the development of NLP tools within and across languages. The objective of a HPOS is to capture the deep detailed information of word grammar which is necessary for many NLP applications like chunking, parsing, morphological analyzer and machine translation etc. Another aim is to standardize the tagsets to achieve cross-linguistic compatibility, reusability and interchangeability.** HPOS tag set is advantageous over flat tag set which captures shallow information. Tag set of higher granularity may actually give better results. Since tagset defined here is prior to any work on the actual tagger, it cannot be known what degree of fine grained tagset will prove optimal for the tagging process. Again it cannot be known in advance how distinctions will prove unfeasible. Therefore the first step must be to make a linguistically ideal tagset. The tagset which we would like to apply to our text in a real world, this ideal tagset, will be the largest within the parameters laid out by hierarchical design principles, on the basis that it is always easier to remove distinctions than to add them. We propose the key design principle here of maximum granularity. EAGLES frame work (Leech and Wilson, 1996) is used as guideline. There was another attempt made towards developing a hierarchical tagset for Indian languages (IL-POST) in 2008 by Microsoft group (Baskran et.al, 2008). But there were many open issues left uncovered. Proposed HPOS tag set resolves the unsolved issues of IL-POST tag set.

- **A Hybrid named entity recognizer/classifier is also developed in order to tag proper nouns in the corpus.** The NER system is built using both *rule based approaches and machine learning approaches.* Two machine learning approaches like HMM (Hidden Markov Model) and CRF (Conditional Random Field) are used here. Similarly another proper noun dictionary of 5000 words has been developed for the task of NER. R*ule based approach for NER* is the first attempt of its kind as par as Kannada is considered. Rule based NER reports a precision of 87%, recall of 90% and F-measure of 87%. HMM, CRF classifiers are used to classify proper nouns in NER system. CRF based classifier is found to be costlier in terms of training time, when more features are considered, but has given 96% classification results when tested with the samples used for training. The same is true for HMM also. But when samples considered are not part of the training data then the recognition accuracy falls down in both CRF and HMM to less than 60%. However performance can be improved by increasing the size of the training data. The CRF classifier has given better results than HMM.

The hypothesis that fine grained tagset is useful for NLP applications like chunking, parsing, morphological analyzer and machine translation etc., is proved by developing morphological analyzer/generator. Standard tagset is necessary for annotation of corpora. The significance of large annotated corpora in the present day NLP is widely acknowledged. Annotated corpora serve as an important tool for investigators of natural language processing, speech recognition and other related areas. Annotated corpora prove to be a basic building block for constructing statistical models for automatic processing of natural languages. POS tagger serves as fundamental building block for NLP research.

Finally, the function of proposed system is to identify and enumerate all the construction types (within the linguistic limits) of a particular language, down to all degree of detail. The output of the morph is represented in a manner such that they could be effectively compared cross-linguistically. This approach of construction labeling would be helpful in developing applications where detailed information is required. The analyzer can be used as a spelling checker as well.

9.2 Aims and Objectives Achieved

The following aims and objectives have been achieved.

- To propose an architecture for building detailed automatic morph syntactic annotator system suitable for Dravidian language family, thereby developing a functional grammar at word level, which gives an algorithmic way of saying, given a word form , whether it is valid form or not.
- To explore the advantages of using a combination of hierarchical tagset and hierarchical dictionary in developing wide coverage, Morphological analyzers/generator by applying FST technology, thereby exploring the challenges in developing morphological analyzer/generator systems for morphological rich language like Kannada. The hypothesis that systems designed using hierarchical tagset performs better than the system using flat tag set in NLP applications like MT and parsing which demand detailed fine grained information is satisfied by implementing morphological analyzer/generator.
- To explore the aspects Kannada language computationally and to improve the availability of NLP resources for Kannada and to facilitate the development of tagged corpora using our annotator tool.
- To explore for very high Recall i.e. all valid word forms to be recognized.

- How an electronic dictionary can be developed by making use of available resources with less manual effort in short time is shown in this work.
- To explore that, Hierarchical tagsets are useful in detailed parsing than flat tagsets.
- Single frame work for morphological generator and analyzer is built here which is bidirectional.

9.3 Future Scope of the Work

- The Morphological analyzer and generator (MAG) system can be used in machine translation applications at later stage. It can be used to develop annotated corpora on which statistical methods can be experimented. The performance of the morph system can be improved by improving the ***size of the dictionary.*** Similarly the performance of NER system with respect to machine learning approaches CRF and HMM can be enhanced by increasing large amount of training data.
- The performance can be further be improved by improving gazetteers list for NER Rule based approach. A **gazetteer** consists of a set of **lists** containing names of entities such as cities, organizations, days of the week,etc,.
- Analyzing the performance using other methods like Support Vector Machines (SVM), Decision tree etc.
- Comparing the results obtained by using different approaches and finding the most accurate approach for it.
- NER systems can be embedded in POS taggers, chunkers.
- Annotated corpora can be developed using the resources developed in this work. Annotated corpora can be used to carry out many statistical experiments.
- The morphological analyzer developed gives morpheme by morpheme by morpheme output and can be used for studing the grammar of the Kannada language tool.
- Our morphological analyzer accepts only valid forms and hence any forminvalid form not accepted is ususlly a spelling mistake form, hence the analyzer can be used as spell ckecker also.

To conclude, the attempted research work on annotation has opened further research avenues for those interested in this field.

PUBLICATIONS

Excellent Paper Award

International Conferences

1. Bhuvaneshwari C Melinamath et al. (2010). "***A Morphological generator for Kannada Based on Finite State Transducers,"*** 3rd IEEE, (ICECT). International Conference on Electronics Computer Technology, Kanyakumari, India 8 – 10 April. 2011. Vol.1, PP. 312-316.
2. Bhuvaneshwari C melinamath (2011). "***A robust Morphological analyzer to capture Kannada noun Morphology,"*** IEEE, International Conference on Future Information Technology (ICFIT). IPCSIT Vol.13 (2011) IACSIT Press, Singapore held at Singapore, during Sept 16-18, 2011. (Excellent Paper Award).
3. Bhuvaneshwari C Melinamath.(2012). *"**Bidirectional Analyzer and Generator Tool for Kannada Nouns,"*** 10th International Conference on Computer Applications (ICCA), University of Computer studies, Yangon. Ministry of Science and Technology, Myanmar. 28-29th February. PP 400-403 2012.
4. Bhuvaneshwari C Melinamath. (2010***). "Generating Kannada Verbal forms, A Computational*** study***".*** 9th International conference on South Asian languages (ICOSAL), held at Patiala Punjabi University, India, during Jan 7-9-2010.

- **International Journal Publications**

1. Bhuvaneshwari C Melinamath. "Rule ***Based Methodology for recognition of Kannada named entities,"*** International journal of Latest trends in Engineering and technology (IJLTET). ISSN: 2278-621X... Vol. 3 Issue 4 March 2014. **Impact Factor 0.685**
2. Bhuvaneshwari C Melinamath. "***Design of Hierarchical Annotator System for Kannada Language,*"** International Journal of Research in Engineering & Technology (IMPACT: IJRET) ISSN (E): 2321-8843; ISSN (P): 2347-4599 Vol. 2, Issue 5, May 2014, 97-110. **Impact Factor 1.548**

3. Bhuvaneshwari C Melinamath. "***IMPROVEMENT OVER IL-POST TAGSET FOR KANNADA,"*** International Journal of Computer Science and Engineering (IJCSE) ISSN (P): 2278-9960; ISSN (E): 2278-9979 Vol. 3, Issue 3, May 2014, 179-186. **Impact factor 3.1323**.
4. Bhuvaneshwari C Melinamath. ***"Design of Morphological generator for Kannada nouns and verbs,"*** International journal of advanced research in computer science and software Engineering (IJARSE). Vol. 4, Issue 4, April 2014, I**SSN: 2277 128X** Impact **factor 2.085.**
5. Bhuvaneshwari C Melinamath. "***Design of Derivational Morphological Analyzer for Kannada Language*"**. **Journal** "**Languages in India**" .Vol. 14:6 June 2014 ISSN 1930-2940.

Appendix A :

Table A1-A.14 Shows Our Hierarchical Tags

Table A1. The Tag List For Adjective

Tag	Description	Example
ADJ-DEM	Demonstrative Adjectives	aa
ADJ-ORD	Adjective Ordinals	OMdaneeya
ADJ-ABS	Adjective absolute	suMdara
ADJ-QNTF	Adjective Quatifiers	Svalpa

Table A2. The List of Tags for Adverbs

Tag	Description	Example
ADV-TIM	Adverb of Time	beegane
ADV-PLA	Adverb of Place	atta
ADV-MAN	Adverb of Manner	oTTaagi
ADV-INTF	Adverb of Intensifiers	atii
ADV-QW	Adverb of Question Type	eeke
ADV-NEG	Adverb of Negative sense	haagalla
ADV-ABS	Adverb absolute	teLLane
ADV-DUP	Adverb of Duplication	Sarasarane
ADV-CONJ	Adverb of Conjunction	aadaagyuu

Table A3. The List of Tags for Interjections, Conjunction and Punctuation Marks

Tag	Description	Example
INTJ	Interjection	Ayyoo
PUN-LEB-SQR	Left Square bracket	[
PUN-LEB-FLR	Left Flower bracket	{
PUN-LEB-PAR	Left Parenthesis bracket	(
PUN-LEB-ANG	Left angular bracket	<
PUN-RIB-SQR	Right square bracket	]
PUN-RIB-FLR	Right Flower bracket	}
PUN-RIB-PAR	Right parenthesis bracket	)
PUN-QUO-SNG	Single quotes	‘
PUN-QUO-DBL	Double quotes	“
PUN-SEP-COL	Separating mark colon	:
PUN-SEP-SEC	Separating mark semi colon	;
PUN-SEP-FUS	Separating mark full stop	.

PUN-SEP-HYP	Separating mark hyphen	-
PUN-SEP-EXC	Separating mark exclamatory	!
PUN-SEP-QUE	Separating question mark	?
PUN-SEP-COMA	Separating mark comma	,
CONJ-COOR	Coordinating Conjunction	Mattu
CONJ-SUB	Subordinating Conjunction	Aadare

Table A4. The List of Tags For Postpositions

Tag	Description	Example
PP-ACC	Postposition Accusative	Annu
PP-DAT	Postposition Dative	Kke
PP-GEN	Postposition Genetive	Ra
PP-SOC	Postposition Sociative	oDane
PP-SIM	Postposition Similarative	aMte
PP-COMP	Postposition Comparative	kkiMta
PP-PUR	Postposition Purpositive	Gooskara
PP-LOCD	Postposition Locative Dative	oLakke
PP-OTH	Postposition Others	Jotege
PP-ABL	Postposition Ablative	iMda

Table A5. List of Tags for Personal Pronoun

Tag	Description
PRO-PER-P1-MFN-SL-[NOM/GEN/DAT/ACC /ABL /PUR /PUR2/COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in singular 1st person
PRO-PER-P2-MFN-SL-[NOM/GEN/DAT/ACC /ABL / PUR /PUR2/COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in singular 2nd person
PRO-PER-P1-MFN-PL-[NOM/GEN/DAT/ACC /ABL / PUR /PUR2/COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in Plural 1st person
PRO-PER-P2-MFN-PL-[NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in plural 2nd person
PRO-PER-P3-M.SL-DIST-[NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in singular 3rd person Masculine distant
PRO-PER-P3-F.SL-DIST- [NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP /	Personal pronoun in

SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	singular 3rd person feminine distant
PRO-PER-P3-N.SL-DIST**-** [NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in singular 3rd person neuter distant
PRO-PER-P3-M.PL-DIST**-[** NOM/GEN/DAT/ACC /ABL / PUR /PUR2/COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in plural 3rd person animate/rational distant
PRO-PER-P3-N.PL-DIST**-[** NOM/GEN/DAT/ACC /ABL / PUR /PUR2/COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in plural 3rd person nonrational distant
PRO-PER-P3-M.SL-PROX**-[** NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in singular 3rd person Masculine proximate
PRO-PER-P3-F.SL-PROX**-[** NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in singular 3rd person feminine proximate
PRO-PER-P3-N.SL-PROX**-[** NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in singular 3rd person neuter proximate
PRO-PER-P3-M.PL-PROX**-[** NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP /SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in plural 3rd person animate/rational proximate
PRO-PER-P3-N.PL-PROX**-[** NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Personal pronoun in plural 3rd person nonrational proximate

Table A6. The List of Tags For Reflexive Pronoun

Tag	Description
PRO-REF-P23 -MFN.SL [NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Reflexive pronoun in singular 2nd, 3rd person
PRO-REF-P23 -MFN.PL [NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Reflexive pronoun in plural 2nd , 3rd person

Table A7. The List of Tags For Interrogative Pronoun

Tag	Description
PRO-INTG-P3.M.SL-[NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Interrogative pronoun in masculine singular 3rd person
PRO-INTG-P3.F.SL-[NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Interrogative pronoun in feminine singular 3rd person
PRO-INTG-P3.N.SL-[NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Interrogative pronoun in neuter singular 3rd person
PRO-INTG-P3.MF.PL-[NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COM P / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Interrogative pronoun in rational plural 3rd person
PRO-INTG-P3.N.PL[NOM/GEN/DAT/ACC /ABL / PUR /PUR2 /COMP / SOC1/SOC2/SIM/LOC /LOCD [CLEM/CLINTR / CLIND/ CLINCL]	Interrogative pronoun in non rational plural 3rd person

Table A8. The List of Tags for Common Countable Noun

Tag	Description	Example
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-NOM	Common countable noun nominative case singular/plural of any gender	huDuga huDugaru
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-ACC	Common countable noun accsative case singular/plural of any gender	huDuganannu huDugarannu
N-COM-COU [M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-DAT	Common countable noun dative case singular/plural of any gender	huDuganige huDugarige
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-ABL	Common countable noun ablative case singular/plural of any gender	huDuganiMda huDugariMda
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-GEN	Common countable noun genetive case singular/plural of any gender	huDugana huDugara
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-LOC	Common countable noun locative case singular/plural of any gender	huDuganalli huDugaralli
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-LOCD	Common countable noun locative dative case singular/plural of any gender	huDuganoLage huDugaroLage
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-SOC1	Common countable noun sociative case singular/plural of any gender	huDuganoDane huDugaroDane
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-SOC2	Common countable noun sociative 2 case singular/plural of any gender	huDuganoTTige huDugaroTTige
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-COMP	Common countable nouncomparative case singular/plural of any gender	huDuganigiMta huDugarigiMta
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-PUR1	Common countable noun purposive case singular/plural of any gender	huDuganigooskar a huDugargooskara
N-COM-COU-[M.SL/F.SL/N.SL/M.PL/F.PL/ N.PL]-PUR2	Common countable noun purposive2 case singular/plural of any gender	huDuganigaagi huDugarigaagi

Table A9. The List of Tags for Common Uncountable Noun

Tag	Description	Example
N-COM-UNC-N.SL-NOM	Common uncountable noun nominative case singular	baMgaara
N-COM-UNC-N.SL-ACC	Common uncountable noun accsative case singular	baMgaaravannu
N-COM-UNC-N.SL-DAT	Common uncountable noun dative case singular	baMgaarakke
N-COM-UNC-N.SL-ABL	Common uncountable noun ablative case singular	baMgaaradiMda
N-COM-UNC-N.SL-GEN	Common uncountable noun genetive case singular	baMgaarada
N-COM-UNC-N.SL-LOC	Common uncountable noun locative case singular	baMgaaradalli
N-COM-UNC-N.SL-LOCD	Common uncountable noun locative dative case singular	baMgaaradoLage
N-COM-UNC-N.SL-SOC1	Common uncountable noun sociative case singular	baMgaaradoDane
N-COM-UNC-N.SL-SOC2	Common uncountable noun sociative 2 case singular	baMgaaradoTTige
N-COM-UNC-N.SL-COMP	Common uncountable noun comparative case singular	baMgaarakkiMta
N-COM-UNC-N.SL-PUR1	Common uncountable noun purposive case singular	baMgaarakkooska ra
N-COM-UNC-N.SL-PUR2	Common uncountable noun purposive2 case singular	baMgaarakkaagi

Table A .10 The List of Tags for Locative Nouns both Time and Place

Tag	**Description**
N-LOC-TIM-NOM [CLEM/CLINTR/CLINDF/ CLINCL	Locative noun denoting time in nominative case
N-LOC-TIM-ACC-[CLEM/CLINTR/CLINDF/ CLINCL	Locative noun denoting time in accusative case
N-LOC-TIM-DAT-[CLEM/CLINTR/CLINDF/ CLINCL	Locative noun denoting time in dative case
N-LOC-TIM-LOC-[CLEM/CLINTR/CLINDF/ CLINCL	Locative noun denoting time in locative case
N-LOC-TIM-ABL-[CLEM/CLINTR/CLINDF/ CLINCL	Locative noun denoting time in ablative case
N-LOC-TIM-GEN-[CLEM/CLINTR/CLINDF/ CLINCL	Locative noun denoting time in genetive case
N-LOC-TIM-PUR1-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting time in purposive case
N-LOC-TIM-PUR2-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting time in purposive case
N-LOC-TIM-COMP-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting time in comparative case
N-LOC-TIM-SOC1-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting time in sociative case
N-LOC-TIM-SOC2-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting time in sociative case
N-LOC-TIM-SIM1-[CLEM/CLINTR/CLINDF/ CLINCL	Locative noun denoting time in similarative case
N-LOC-PLA-NOM-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in nominative case
N-LOC-PLA-ACC-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in accusative case
N-LOC-PLA-DAT-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in dative case
N-LOC-PLA-ABL-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in ablative case
N-LOC-PLA-GEN-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place

	in genetive case
N-LOC-PLA-LOC-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in locative case
N-LOC-PLA-LOCD [CLEM/CLINTR/CLINDF /CLINCL	Locative noun denoting place in locative dative case
N-LOC-PLA-COMP[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in comparative case
N-LOC-PLA-SOC1-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in sociative case
N-LOC-PLA-SOC2-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in sociative case
N-LOC-PLA-SIM-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in similarative case
N-LOC-PLA-PUR1-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in purposive case
N-LOC-PLA-PUR2-[CLEM/CLINTR/CLINDF/CLINCL	Locative noun denoting place in purposive case

Table A. 11. The List of Tags Noun Cardinals

Tag	**Description**
N-CARD-HUM-NOM-[CLEM/CLINTR/CLINDF/ CLINCL	Noun cardinal for human in nominative case
N-CARD-HUM-ACC-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in accusative case
N-CARD-HUM-DAT-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in dative case
N-CARD-HUM-ABL-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in ablative case
N-CARD-HUM-GEN-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in genetive case
N-CARD-HUM-LOC-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in locative case
N-CARD-HUM-LOCD-[CLEM/CLINTR/CLINDF /CLINCL	Noun cardinal for human in locative dative case
N-CARD-HUM-SOC1-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in sociative case
N-CARD-HUM-SOC2-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in sociative case
N-CARD-HUM-PUR-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in purposive case
N-CARD-HUM-COMP-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in comparative case
N-CARD-HUM-SIM1-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for human in similarative case
N-CARD-NHU-NOM-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for nonhuman in nominative case
N-CARD-NHU-ACC-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for non human in accusative case
N-CARD-NHU-DAT-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for nonhuman in dative case
N-CARD-NHU-ABL-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for nonhuman in ablative case
N-CARD-NHU-GEN-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for nonhuman in genetive case

N-CARD-NHU-LOC-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for nonhuman in locative case
N-CARD-NHU-LOCD-[CLEM/CLINTR/CLINDF/ CLINCL	Noun cardinal for nonhuman in locative dative case
N-CARD-NHU-COMP-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for nonhuman in comparative case
N-CARD-NHU-SOC-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for nonhuman in sociative case
N-CARD-NHU-PUR-[CLEM/CLINTR/CLINDF/CLINCL	Noun cardinal for nonhuman in purposive case

Table A. 12. The List of Tags for Non Finite Verbs

Tag	**Description**
V-NF-TR/IN/BI-CAU-REF-PRES-CJP [1/2/3/4/5/6/7/8/9/10/11/12]-AFF	Non-finite verb in present conjunctive affirmative
V-NF-TR/IN/BI-CAU-REF-PRES-CJP [1/2/3/4/5/6/7/8/9/10/11/12]-AFF-ITER	Non-finite verb in present conjunctive affirmative iterative aspect
V-NF-TR/IN/BI--CAU-REF-PST-CJP [1/2/3/4/5/6/7/8/9/10/11/12]-AFF	Non-finite verb in Past conjunctive affirmative
V-NF-TR/IN/BI--CAU-REF-CJP [1/2/3/4/5/6/7/8/9/10/11/12]-AFF-PERF	Non-finite verb in Past conjunctive affirmative perfective
V-NF-TR/IN/BI--CAU-REF- [1/2/3/4/5/6/7/8/9/10/11/12]-AFF-COND	Non-finite verb in conditional sense
V-NF-TR/IN/BI--CAU-REF-CONC [1/2/3/4/5/6/7/8/9/10/11/12]-AFF	Non-finite verb in concessive affirmative
V-NF-TR/IN/BI--CAU-REF-PST-CJP [1/2/3/4/5/6/7/8/9/10/11/12]-NEG	Non-finite verb in Past conjunctive negative
V-NF-TR/IN/BI--CAU-REF-PST-CJP [1/2/3/4/5/6/7/8/9/10/11/12]-NEG-COND	Non-finite verb in Past conditional negative
V-NF-TR/IN/BI--CAU-REF-PST-CJP [1/2/3/4/5/6/7/8/9/10/11/12]-NEG-CONC	Non-finite verb in Past concessive negative

Table A. 13 Modal Auxilaries

NO	TAG
1	CAP
2	NCAP
3	PROH
4	NPRO
5	PERM
6	NPER
7	PERM
8	NPER

9	PASS
10	INCP
11	OBL
12	NEG

Table A14. The List of Tags for Finite Verbs

Tag	**Description**
V-FI-TR/INTR/BI-CAU-REF-AFF-PST-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Past
V-FI-TR/INTR/BI-CAU-REF-AFF-PRES-[M.SL/F.SL/N.SL/MF.PL [N.PL/P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL]	Finite verb in affirmative Present
V-FI-TR/INTR/BI- CAU-REF-AFF-FUT-[M.SL/F.SL/N.SL/MF.PL [N.PL/P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL]	Finite verb in affirmative Future
V-FI-TR/INTR/BI-CAU-REF-AFF-IMP-[P2.MFN.SL/P2.MFN.PL]	Finite verb in affirmative Imperative
V-FI-TR/INTR/BI-CAU-REF-AFF-PIMP-[P2.MFN.SL/P2.MFN.PL]	Finite verb in affirmative Polite Imperative
V-FI-TR/INTR/BI-CAU-REF-AFF-HOR-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Hortative
V-FI-TR/INTR/BI-CAU-REF-AFF-CAP-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Capablative
V-FI-TR/INTR/BI-CAU-REF-AFF-CNTG-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Contigent
V-FI-TR/INTR/BI-CAU-REF-AFF-NCAP-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Non capablative
V-FI-TR/INTR/BI-CAU-REF-AFF-PERM	Finite verb in affirmative

	Permissive
V-FI-TR/INTR/BI-CAU-REF-AFF-PROB	Finite verb in affirmative Probablative
V-FI-TR/INTR/BI-CAU-REF-AFF-OPT	Finite verb in affirmative Optative
V-FI-TR/INTR/BI-CAU-REF-AFF-PASS-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Passive Past
V-FI-TR/INTR/BI-CAU-REF-NEG-PST-	Finite verb in negative Past
V-FI-TR/INTR/BI-CAU-REF-PRES-DUR-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Present durative
V-FI-TR/INTR/BI-CAUR-REF-PROH	Finite verb in affirmative Prohibitive
V-FI-TR/INTR/BI-CAU-REF-PRES-PERF-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Present perfective
V-FI-TR/INTR/BI-CAU-REF-PRES-DUR-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Present durative
V-FI-TR/INTR/BI-CAU-REF-PST-DUR-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Past durative
V-FI-TR/INTR/BI-CAU-REF-PST-PERF-[M.SL/F.SL/N.SL/MF.PL/N.PL /P1.MFN.SL/P2.MFN.SL/P3.F.SL/P3.M.SL/P3.N.SL/P3.MF.PL/P3.N.PL/P1.MFN.PL /P2.MFN.PL]	Finite verb in affirmative Past perfective
V-FI-TR/INTR/BI-CAU-REF-NEG-PST-DUR	Finite verb in negative

	Past durative
V-FI-TR/INTR/BI-CAU-REF-NEG-FUT	Finite verb in negative Future
V-FI-TR/INTR/BI-CAU-REF-NEG-PST-PERF	Finite verb in negative past perfective
V-FI-TR/INTR/BI-CAU-REF-NEG-PRES	Finite verb in negative Present
V-FI-TR/INTR/BI-CAU-REF-NEG-PRES-DUR	Finite verb in affirmative Present durative

Appendix B :Sample Morph Output

```
1: maaDalu
   1: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : null -
absolute - infinitive
2: maaDikoMDu
   1: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : null -
past_verbal_participle - reflexive -
past_verbal_participle -
3: maaDitu
   1: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : null -
absolute - past - p3_n_singular - null -
4: maaDiddaare
   1: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : null -
perfective_present - p3_mf_plural - null -
5: maaDalpaTTu
   1: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : null -
absolute - infinitive - null - passive -
past_verbal_participle -
6: maaDikoMDevu
   1: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : null -
past_verbal_participle - reflexive - absolute - past -
p1_mfn_plural - null -
8: maaDikoLLalu
   1: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : null -
past_verbal_participle - reflexive - absolute -
infinitive -
9: maaDikoMDirabeeku
   1: [maaDu: N-COM-COU-NS-NOM-NULL||V-] : null -
past_verbal_participle - reflexive -
past_verbal_participle - asp_aux_iru_or_perfective -
absolute - infinitive - null - compulsive - null -
```

Appendix C :

Sample Text Tagged With HPOS Tagset

tamma<PRO-REF-P23.MFN.PL-GEN ->sampradaayavannu<N-COM-COU-N.SL-ACC->elleDe<ADV-PLA>habbisuva<V-IN-CAU-RP>uddes`adiMda<N-COM-COU-N.SL-ABL> dehali<N-PRP-LOC> vaDaiko<N_N-PRP-ORG>taMDa<N-COM-COU-N.SL-NOM>japaan<N-PRP-LOC>habbada<N-COM-COU-N.SL-GEN>nepavaagi<ADV-DE_N>beMgaLuurige<N-PRP-LOC>baMdittu<V-IN-PST-N.SL>. japaanina<N-PRP-LOC>saMpradaayika<ADJ-DESC->s`ailiya<N-COM-COU–N.SL -NOM>TaikooDram <?>naadavannu<N-COM-COU-N.SL-NOM>elleDe<ADV-PLA>pasarisuva<V-TR-CAU-FUT-RP> bayake<N-COM-COU-N.SL-NOM->ivaradu<PRP-PER-P3-M.SL>adu.<?><.><PUN-SEP-FUS> cunaavaNe<N-COM-COU-N.SL-NOM> prakriyeyalli <N-COM-COU-N.SL-LOC >paris`ishTa<ADJ ABS>jaati<N-COM-COU-N.SL-NOM>haagu<CONJ-COOR>paris`ishTa<ADJ-ABS>vargagaLa<N-COM-COU-N.PL-GEN>miisalu<ADJ-ABS>saulabhya<N-COM-COU-N.SL-NOM>paDeyalu<V-TR-INF> bijepi<N-PRP-ORG> sadasyaree<N-COM-COU-M.PL-NOM-EMP>suLLu<N-COM-COU-N.SL-NOM>jaati<N-COM-COU-N.SL-NOM>pramaaNa<N-COM-COU-N.SL-NOM>patra<N-COM-COU-N.SL-NOM> niDiddaare<V-TR-PST-M.PL>ennuva<V-IN-FUT-RP>samaaja<<N-COM-COU-N.SL-NOM>kalyaNa<N-COM-UNC-N.SL-NOM>sachiva<N-COM-COU-M.SL-NOM> eh.aMjaneeya<N-PRP-PER-M.SL-NOM>avara<PRO-PER-P3-M.SL-NOM>keelike<N-COM-COU-N.SL-NOM><,><PUN-SEP-COMA>vidhana<N-COM-COU-N.SL-NOM> parishatnalli <N-COM-UNC-N.SL-NOM >soomavaara <N-LOC-TIM-ABS-NOM>addallakke<N-COM-UNC-N.SL-NOM>kaaraNavaayitu<N-COM-COU-N.SL-NOM→V+ aagu+PST-N.SL><.><PUN-SEP-FUS>niirashTee<N-COM-UNC-CLIT-EMP>,alla <ADV-NEG><,><PUN-SEP-COMA> isrelanalli <N-PRP-LOC>maNNu<N-COM-UNC-N.SL-NOM>kuuDa <CONJ-COOR>amulya<ADJ-ABS>ardhakkiMta<N-CARD-COMP> heccu<ADJ-QNTFF>prades`a<N-COM-COU--N.SL-NOM>marabhumi<N-COM-COU-N.SL-NOM>aadare<CONJ-COOR>allina<N-LOC-PLA-GEN>kRshi<N-COM-UNC-N.SL-NOM->utpannagaLa<N-COM-COU-N.PL-NOM>iLuvari<N-COM-UNC-N.SL-NOM>yaavudee<PRO-INTG-P3-N.SL-NOM>des`akkiMta<N-COM-COU-N.SL-NOM>heccu <ADJ-QNTF><.><PUN-SEP-FUS>aparuupakke<N-COM-UNC-N.SL-DAT->siguva<V-IN-FUT-RP>maNNannu<N-

COM-UNC-N.SL-ACC>taMdu<V-TR-CJP><,><PUN-SEP-COMA>vivechaneyiMda<N-COM-UNC-N.SL-NOM>baLasi<V-TR-CAU-PST-CJP>vyavasaaya<N-COM-UNC-N.SL-NOM->naDeyuttide<V-IN-PRES-N.SL><.>

Appendix D: Sample Train Data for HMM

[NEL]navadehali[OTHER]:: [OTHER]tamma [OTHER]sarkaarada[OTHER]meele[OTHER] baMdiruva [OTHER]bhrashTaacaara[OTHER]aaroopavannu[NET]soomavaara[NEO]kaaMgres [NEL]beMgaLuuru [NEL]dehali[OTHER]kaaryakaariNiyalli [OTHER]taLLihaakiruva [NED]pradhaani [NEP]manamoohan [NEP]siMg[OTHER],OTHER]saarvajanika[OTHER]ksheetradalliruva[NEO]haikama aMD[OTHER]eccarike[NEP]jaganmoohan [NEP]reDDi [OTHER]paaLaya [OTHER]seerida [OTHER]s`aasakara [NEP]jagange [OTHER]jaamiinu [OTHER]niraakaraNe.

[NEL]haidaraabaad : [OTHER]akrama [OTHER]aasti [OTHER]saMpaadane [OTHER]prakaraNadalli [OTHER]baMdhanakkoLagaagiruva [NEP]vai.es.jaganmoohan [NEP]reDDi (jagan) [OTHER]avaru [OTHER]eraDanee [OTHER]baari [OTHER]sallisidda [OTHER]jaamiinu [OTHER]arjiyannu [NEO]sibiai [NEO]koorT [NET]s`ukravaara [OTHER]niraakariside.[OTHER]upa [OTHER]cunaavaNe [OTHER]pracaarakke [OTHER]teraLalu [OTHER]avakaas`a [OTHER]niiDabeekeMdu [OTHER]koori [NEP]jagan [OTHER]jaamiinu [OTHER]keeLiddaru.jagan [OTHER]avaraMtha [OTHER]prabhaavi [OTHER]vyaktige [OTHER]jaamiinu [OTHER]niiDidare [OTHER]avaru [OTHER]prakaraNada [OTHER]meele [OTHER]prabhaava [OTHER]biiruva [OTHER]saadhyate [OTHER]iruttade [OTHER], [OTHER]haagaagi [OTHER]avarige [OTHER]jaamiinu [OTHER]niiDabaaradu [OTHER]eMdu [NEO]sibiai [OTHER]koorittu.[NET]9 [NET]tiMgaLa [OTHER]hiMdeyee [OTHER]aaraMbhavaada [OTHER]tanikheyalli [NEP]jagan [OTHER]madhyapravees`isilla .[OTHER] [OTHER]avaru [OTHER]muMdina [OTHER]tanikheguu [OTHER]sahakarisuttaare[OTHER]. [OTHER]haagaagi [OTHER]avarige [OTHER]jaamiinu [OTHER]niiDabeeku [OTHER]eMdu [NEP]jagan [OTHER]para [OTHER]vakiilaru [OTHER]vaadisidaru [OTHER].[OTHER]bhrashTaacaara [OTHER]toDeduhaaki[OTHER]paaradars`aka

BIBLIOGRAPHY

1 A. Borthwick, J. Sterling, E. Agichtein, and R. Grishman. *Exploiting diverse knowledge sources via maximum entropy in named entity recognition*. Proc. of the Sixth Workshop on Very Large Corpora, pp. 152–160, 1998.

2 A Hardie. *Developing a tagset for automated part-of speech tagging in Urdu*. Proc. of the Corpus Linguistics 2003 conference, 16, 2003.

3 A Hardie. *The computational Analysis of Morphosyntactic Categories in Urdu*. PhD Thesis, Department of Linguistics, Feb 2004.

4 Andrew McCallum and Wei Li . *Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web Enhanced Lexicons*. doi=10.1.1.3.6795.pdf,2000.

5 A Kumano and H Hirakawa. *Building a dictionary from parallel texts based on linguistic and statistic information*. Proc. of International Conference on Computational Linguistics, pages 76–81, Kyoto, 1994

6 Akshar Bharathi, Rajeev Sangal and Sushma M Bendre. *Some Observations Regarding Corpora of Indian Languages*. Proc.of International Conference on Knowledge-Based Computer Systems (KBCS-98), NCST, Mumba**i,** 17-19 Dec 1998.

7 Alicia et al. *A compiler for morphological analysers and generators based on finite-state transducers*. Department of Linguistics, Spain

8 A. L. Berger. *The improved iterative scaling algorithm:* A gentle introduction. 1997

9 Alireza Mansouri, Lilly Suriani Affendey, Ali Mamat. *Named Entity Recognition Approaches, International J. of Computer Science and Network Security (IJCSNS)*. Vol.8 issue 2, February 2008.

10 Amarappa and S V Sathyanarayana. *Named Entity Recognition and Classification in Kannada Language*. ISSN 2277-1956/V2N1-281-289, 2012.

11 Andrei Mikheev, Marc Moens and Claire Grover. *Named Entity Recognition without Gazetteers*. Proc. of *EACL 1999.*

12 Aniket Dalal, Kumar Nagaraj, Uma Sawant, and Sandeep Shelke. *Hindi part-of speech tagging and chunking, A maximum entropy approach*. Proc. of NLPAI Machine Learning Contest, IIIT, Hyderabad, India, 2006.

13 Anil Kumar Singh. *Named Entity Recognition for South and South East Asian Languages*. Proc. of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India, pp. 5–16, January 2008.

14 Animesh Nayan, B Ravi Kiran Rao, Pawandeep Singh, Sudip Sanyal and Ratna Sanyal. *Named Entity Recognition for Indian Languages*. Proc. of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India, pp. 97–104, January 2008

15 Ashara, Masayuki and Matsumoto. *Japanese Named Entity Extraction with Redundant MorphologicalAnalysis*. Proc. of Human Language Technology conference, North American chapter of the Association for Computational Linguistics, 2003.

16 Asif Ekbal, Rejwanul Haque, Sivaji Bandyopadhyay. *Named Entity Recognition in Bengali: A Conditional Random Field Approach*. http://wwwaclweborg/anthology-new/I/I08/I08-2077 pdf . 2008.

17 Asif Ekbal and Samiran Mandal. *PoS Tagging using HMM and Rule based Chunking.* Proc. of International Joint Conference on Artificial Intelligence workshop on Shallow Parsing for South Asian Languages, Language Technologies Research Centre, IIIT, Hyderabad, 2007.

18 Asif Ekbal and Sivaji Bandyopadhyay. *Named Entity Recognition using Support Vector Machine, A Language Independent Approach*. International J. of Electrical and Electronics Engineering vol.4, issue 2, 2010.

19 Asif Ekbal, Rejwanul Haque, Amitava Das, Venkateswarlu Poka and Sivaji Bandyopadhyay. *Language Independent Named Entity Recognition in Indian Languages.* Proc. of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India, pp 33–40, January 2008.

20 Asif Ekbal and S. Bandyopadhyay. *Named entity recognition in Bengali: A Conditional random field*. Proc. of ICON, pp 123–128, 2008.

21 B Sasidhar, P M Yohan, Dr A Vinaya Babu3, Dr A Govardhan. *A Survey on Named Entity Recognition in Indian Languages with particular reference to Telugu*. International J. of Computer Science(IJCSI), Vol. 8, Issue 2, March 2011.

22 Bask ran et al,. *Designing a common pos-tagset framework for Indian languages,* Proc. of 6th Workshop on Asian Language Resources, Language

Technologies, Research Centre, IIIT, Hyderabad, 2008.

23 B. D. M, M. Scott, S. Richard, and W. Ralph. *A High Performance Learning Name-finder*. Proc.of the fifth Conference on Applied Natural language Processing. pp. 194–201, 1997.

24 Beard, R. *Lexeme-Morpheme Base Morphology.* A General Theory of Inflection and Word Formation. State University of New York Press , 1995.

25 Beesley, K. and L. Karttunen. *Finite State Morphology*. Stanford, CA: CSLI Publications, 2003.

26 Bharati, Akshar, Vineet Chaitanya and Rajeev Sangal. *Natural Language Processing. A Paninian Perspective.* Prentice-Hall of India. New Delhi, 1995.

27 Bick, Eckhard . *A Named Entity Recognizer for Danish*. Proc. of the Conference on Language Resources and Evaluation, 2004.

28 Buckwalter T. *Arabic Morphological Analyzer Version 1.0.* Linguistic Data Consortium, University of Pennsylvania, LDC Catalog No.: LDC2002L49, 2002.

29 Cartic Ramakrishnan, Krys J Kochut and Amit P Sheth. *A Framework for Schema-Driven Relationship Discovery from Unstructured text*. ISWC, LNCS 4273, pp. 583 – 596, 2006.

30 Cavalli-Sforza, V. Soudi A, and Mitamura T. *Arabic Morphology Generation Using a Concatenative Strategy*. Proc.of NAACL 2000. Seattle, WA 2000.

31 Centro Cultural de Belem LISBON.*Beyond Named Entity Recognition Semantic labelling for NLP tasks 4th International Conference OnLanguage Resources And Evaluation*. Portugal, 2004.

32 Chanod, Jean-Pierre. *Finite state composition of French verbmorphology.* Technical report MLTT-005, Xerox Research Center Europe, Meylan, France,1994

33 Chanod, Karttunen, Lauri, Jean-Pierre, Gregory Grefenstette, and Anne Schiller. *Regular expressions for language engineering*. Natural Language Engineering,2(4)305-328,1996.

34 Creutz M. *Unsupervised segmentation of words using prior distributions of morph length and frequency*. Proc. of the Association for Computations Languages (ACL03), pp. 280–287, Sapporo, Japan, 2003.

35 Dhanalakshmi V, Anandkumar M, Rekha R U, Arunkumar C, Soman K P, Rajendran S. *Morphological Analyzer for Agglutinative Languages Using Machine Learning Approaches*. Proc. of International Conference on Advances in Recent Technologies in Communication and Computing, IEEE, pp. 433-435, 2009.

36 Darroch J and Ratcliff D.*Generalized Iterative Scaling for log-linear models*. Ann Math Statistics, 43, pp.1470-1480, 1972.

37 D Somars Jones. *Bilingual vocabulary estimation from noisy parallel corpora using Variable bag estimation*. International J. JADT, vol 19, No 1:pp 255–262, 1995.

38 DNS Bhat. *Kannada vaakyagaLa rachane*. Srishaktipress, Mysore.

39 D. N. Shankara Bhat. *KannaDada sarvanaamagaLu.* Published by Bhasha Prakashana, Mysore. May 5, 2003

40 Darvinder kaur, Vishal Gupta. *A survey of Named Entity Recognition in English and other Indian Languages*. International J. of Computer Science Issues, (*IJCSI)* Vol 7, Issue 6, November 2010.

41 Fleischman Michael. *Automated sub categorization of named entities*. Proc. of Conference of the European Chapter of Association for Computational Linguistic, pp 25–30, 2001.

42 Garside R, G. Leech and G. Sampson (eds). *The CLAWS Word-tagging System. In: The Computational Analysis of English: A Corpus-based Approach*. London: Longman.

43 Goldberg D. E.*Genetic Algorithm in Search, Optimization and Machine Learning*. Addison-Wesley 1989.

44 Goldsmith and Gaussier. *Unsupervised learning of derivational morphology from inflection lexicons*. Proc.of the ACL Workshop, pp. 24–30, 1999.

45 John Goldsmith. *Unsupervised learning of the morphology of a natural language Computational Linguistics*. pp. 153–193, 2001.

46 John McCarthy. *A prosodic theory of non concatenative morphology.* Linguistic Inquiry, 12(3):373–418, 1981.

47 Greene B and Rubin G M. *Automatic grammatical tagging of English, Providence, RI: Department of Linguistics*, Brown University, 1981.

48 Grishman. *The nyu system for muc-6*. Proc. of the Sixth Message Understanding Conference (MUC-6), pp. 167–195, Fairfax, Virginia. 1995.

49 Gumwon Hong, Min-Jeong Kim, Do-Gil Lee and Hae-Chang Rim . *A Hybrid Approach to English-Korean Name Transliteration*. Proc. of the Named Entities Workshop, ACL-IJCNLP, pp 108–111,2009.

50 Habash Nizar. *Large scale lexeme based Arabic morphological generation*. Proc. of the Treatment Automatique du Language Natural (TALN-04). Fez, Morocco, 2004.

51 Hammarstrom. *A Naive Theory of Affixation and an Algorithm for Extraction*. SIGPHON Proc. of the Eighth Meeting of the ACL Special Interest Group on Computational Phonology and Morphology at HLT-NAACL, New York City, USA,pp.79-88 June 2006.

52 Harold Spencer. *Kanarese grammar*. Wesltyan Mission Press, 1950.

53 Harold Schiff man. *The grammar for Spoken Kannada*. 1979.

54 Himanshu Agrawal and Anirudh Mani. *Part of speech tagging and chunking with*
Conditional random. Proc. of the NLPAI Machine Learning Contest, IIIT Hyderabad, India, 2006.

55 Hop croft J.E and J D Ullaman. *Introduction to automata theory languages and computation*. Reading MA: Addition Wesly, 1979.

56 Hsin Chen and Shihchung tsai. *Named entity extraction for information retrieval*. Proc. of the *HLT-NAACL*, pp 8-15, 2003.

57 James Mayfield and Paul McNamee and Christine Piatko. *Named Entity Recognition using Hundreds of Thousands of Features*.

58 John Chen et. Al. *Towards automatic generation of natural language generation systems*. Proc. of the 18th International Conference on Computational Linguistics, New York City, USA, 2000.

59 K. Koskenniemi. *Two-level morphology: A general computational model for word form Recognition and production Master's thesis*. Department of General Linguistics, Helsinki University, 1983.

60 Kaplan and Kay M. 1994. *Regular models of phonological rule systems.* Computational Linguistics, 20(3):331–378.

61 Karttunen, Lauri. *Finite-state constraints*. International J. Goldsmith, ed, The Last
Phonological Rule, Chicago, Illinois,: University of Chicago Press,1993.

62 Karttunen, Lauri. *Finite-state lexicon compiler*. Tech. Rep. ISTL-NLTT-1993-04-02, Xerox Palo Alto Research Center, Palo Alto, CA, 1993.

63 Karttunen, Lauri, Ronald M. Kaplan, and Annie Zaenen. *Two-level morphology with composition*. Proc. of the *COLING'92*, pp. 141–148. Nantes, France, 1992.

64 Karttunen, Lauri. *The replace operator*. Proc. of the ACL-95 Cambridge, MA. cmplg/9504032. 1995.

65 Karthik Gali, Harshit Surana, Ashwini Vaidya, Praneeth Shishtla and Dipti Misra Sharma. *Aggregating Machine Learning and Rule Based Heuristics for Named Entity Recognition.* Proc. of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India pp 25–32, 2008.

66 D. Bernhard. 2006. *Unsupervised morphological segmentation based on segment predictability and word segment alignment*. In PASCAL Challenge Workshop on Unsupervised Segmentation of Words into Morphemes.

67 Kavi Narayana Murthy. *Dictionary and computational tools*. J. Linguistics today, vol. 11, No 1, 1997.

68 Kava Narayana Murthy. *A network and process model for morphological analysis/ generation*. Proc. of International Conference on South Asian Languages, Punjabi University, Patiala, India, 9-11 January, 1999.

69 Kay M and Roscheisen M. *Text-Translation Alignment*. Journal Computational Linguistics, Vol. 19, No 1, pp 121-142, 1993.

70 Martin Kay. *Nonconcatenative finite-state morphology*. Proc. of the Third Conference the European Chapter of the Association for Computational Linguistics, pages 2–10, 1987.

71 Kaplan, Ronald M. and Martin Kay. *Regular Models of Phonological Rule Systems*. Computational Linguistics, 20(3):331-378, 1994.

72 K. Beesley. *Arabic finite-state morphological analysis and generation.* Proc of the 16th International Conference on Computational Linguistics (COLING-96), volume 1, pp. 89–94, Copenhagen, Denmark, 1996.

73 Kenneth R Beesley. *Arabic morphology using finite state operations*. 1983.

74 Kittel. F. *A Grammar of Kannada Language in English.* Basel Mission Book and Tract Depository, 1903.

75 Kumano, A and Hirakawa. H.*Building an MT Dictionary from Parallel Texts Based on Linguistics and Statistic Information*. Proc. of the International Conference on Computational Linguistics, pp. 76-81, Kyoto1994.

76 Kiraz, G. *Multi-tape Two-level Morphology: A Case study in Semitic Non-Linear Morphology*. Proc. of COLING'94, Vol. 1, pp. 180-186, 1994.

77 Lafferty J, McCallum A and Pereira F. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. Proc. of the Eighteenth International Conference on Machine Learning, pp. 282-289, 2001.

78 Lawrence R Rabiner. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. Proc. of the IEEE, 77 (2), pp. 257-286, February 1989.

79 Leech G and Wilson.*A Recommendations for morph syntactic annotation of corpora EAGLES Technical Report*. 1996.

80 McCarthy, John J.*The Foundations of Optimality Theory*. Cambridge, England: Cambridge University Press, 2002.

81 Michael Collins and Yoram Singer. *Unsupervised Models for Named Entity Classification*. Proc. of the Joint SIGDAT Conference onEmpirical Methods in Natural Language Processingand Very Large Corpora 1999.

82 Michael Collins. *Ranking algorithms for named entity extraction: Boosting and the voted perception*. Proc. of 40th Annual Meeting of the Association for Computational Linguistics (ACL), pp 489–496, and 2002.

83 M Haruno and Yamazaki. *High precision bilingual text alignment using statically and dictionary information*. Proc. of Annual Conference of the Association for Computational Linguistics, pp. 131–138, Kyoto, 1996.

84 Mohri M. *Finite-state transducers in language and speech processing.* Computational Linguistics, 23(2), pp. 269–311, 1997.

85 N. Verma and P. Bhattacharyya. *Automatic Lexicon Generation through WordNet, Global WordNet Conference (GWC-2004)*. Czech Republic, January 2004.

86 S. Dasgupta and V. Ng. 2007. *Unsupervised word segmentation for Bangla*. Proc. of the ICON, pages 15-24.

87 P. Srikanth and Kavi. Narayana Murthy. *Named Entity Recognition for Telugu*. Proc. of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India, pp. 41–50, January 2008.

88 Patrick Hanks. *Compiling a Monolingual Dictionary for Native Speakers*. Proc. of the Conference on Dictionaries more than words, at Faculty of social Sciences, University of Liublijana, feb-6 2009.

89 Pietra S D, Pietra V D and Lafferty J. *Inducing features of random fields*.Technical Report CMU-CS, pp. 95-144, Carnegie Mellon University, 1995.

90 Pinto D, McCallum A, Wei X And Croft W B. *Table extraction using conditional random fields*. Proc. of the ACM SIGIR, 2003.

91 Pramod Kumar Gupta, Sunita Arora. *An Approach for Named Entiy Recognition System for Hindi: An Experimental Study.* Proc. of ASCNT-2009, CDAC, Noida, India, pp. 103 – 108 , 2009.

92 Praneeth M Shishtla, Prasad Pingali, Vasudeva Varma. *Character n-gram Based Approach for Improved Recall in Indian Language*. Proc. of the NER IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India, pp. 67–74,2008.

93 Praveen Kumar P, Ravi Kiran V. *A Hybrid Named Entity Recognition System for South Asian Languages*. Proc.of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad, India, pp. 83–88, January 2008.

94 Rahul Gupta. *Conditional Random Fields.* http://wwwitiitbacin/~grahul/mainpdf

95 Ramasamy Veerappan, Antony P J, S Saravanan and Dr. Soman K P. *Article: A Rule based Kannada Morphological Analyzer and Generator using Finite State Transducer*. International J. of Computer Applications 27(10):45-52, August 2011.

96 Ralph Grishman et al,. *Message Understanding Conference - 6: A Brief History*. Proc. of the 16th International Conference on Computational Linguistics, 1996.

97 Ramasamy and Dr. Soman K P. *Article: A Rule based Kannada Morphological*

Analyzer and Generator using Finite State Transducer. International Journal of Computer Applications 27(10):45-52, August 2011.

98 Riaz. Kashif. *Named Entities Workshop*. Proc. of the Association for Computational Linguistics ACL, Uppsala, Sweden., pp. 126–135, 16 July 2010.

99 Roche, E and Y Schabes. *Introduction, In finite state language processing*. Ed . By E. Roche and Y Schabes, 1-65. Cambridge, Mass, MIT Press, 1997.

100 Rohini Srihari, Cheng Niu and Wei Li. *A Hybrid Approach for Named Entity and Sub-Type Tagging SBIR grant F30602-98-C-0043 from Air Force Research Laboratory (AFRL) /IFED*. 2008.

101 S. Pandian, K. A. Pavithra, and T. Geetha. *Hybrid Three-stage Named Entity Recognizer for Tamil, INFOS-2008*. March Cairo-Egypt, 2008.

102 S.N. Sridhar. *Descriptive Grammar for Kannada Rout ledge*. New Fetter lane London Ec48EE, 1990.

103 Sandipan Dandapat and Sudeshna Sarkar. *Part of speech tagging for Bengali with Hidden Markova model*. In NLPAI Machine Learning Contest 2006.

104 Santorini .B. *Part of Speech tagging guidelines for the Penn Treebank Project Technical report MS-CIS 47-90*. Department of Computer and Information Science, University of Pennsylvania 1990.

105 Sha .F and Pereira. F. *Shallow parsing with conditional random fields.* Proc. of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Edmonton, Canada, pp.134-141, 2003.

106 Shalini urs and T Vikram. *Development of Prototype Morphological Analyzer for the South Indian Language of Kannada*. Proc. of 10th International Conference on Asian Digital Libraries, ICADL, Hanoi, Vietnam, December 10-13, 2007.

107 Shambhavi et al,. *Kannada Morphological Analyser and Generator Using Trie*. International J. of Computer Science and Network Security (IJCSNS), VOL.11 No.1, January 2011.

108 Shilpi Srivatsava, Mukund Sangalikar, and D.C. Kothari.*Named entity recognition System for Hindi language*. International j. of Computational Linguistics vol. (2), pp 10–23, 2011.

109 Shreen Khoja. *A tagset for the morphosyntactic tagging of Arabic*. Proc. of Corpus Linguistics, Lancaster University, Lancaster, UK, March 2001.

110 Shrivastav M, Melz R, Singh S, Gupta K and Bhattacharyya P. *Conditional Random Field Based POS Tagger for Hindi*. Proc. of the MSPIL, Bombay,63-68, 2006.

111 Sivaji Bandyopadhyay, Asif Ekbal, and Debasish Halder. *Hmm based pos tagger and rule based chunker for Bengali*. Proc. of NLPAI Machine Learning Contest, 2006.

112 Sudha Morwal, Nusrat Jahan. *Named Entity Recognition Using Hidden Markov Model (HMM): An Experimental Result on Hindi, Urdu and Marathi Languages.* ISSN 2277-128X /V2N1-671-675, Vol.3, Issue 4, International J. of Advanced Research in Computer Science and Software Engineering.

113 Sujan Kumar Saha, Sanjay Chatterji, Sandipan Dandapat, Sudeshna Sarkar, Pabitra Mitra. *A Hybrid Approach for Named Entity Recognition in Indian Languages*. Proc. of the IJCNLP-08 Workshop on NER for South and South East Asian Languages, Hyderabad,India, pp. 17–24, January 2008.

114 S.N. Sridhar. *A Descriptive Grammar of Kannada language.* Routledge University, London.

115 Shereen Khoja et al. 2001. *A tagset for the morphosyntactic tagging of Arabic.* Proc. of the Corpus linguistics 2001. pp. 341-353, Lancaster,UK, 2001.

116 T. Utsuro et al. *Bilingual text matching using bilingual dictionary and statistics.* Proc. of the International Conference on Computational Linguistics, pp. 1076–1082, Kyoto, 1994.

117 Tokunaga et al. *Automatic thesaurus construction based on grammatical relation*. Proc. of the IJCAI-95, 1995.

118 UmaMaheshwara and G Parameshwari. O*n the description of morphological data for morphological analyzers and generators: A Case of Telugu, Tamil and Kannada*. CALTS, University of Hyderabad, 2010.

119 Upadhyaya and Krishnamurthy . *Impersonal construction.* 1972: 154.

120 Vijay and Shobha. *Tamil morphological analyzer*. http://nrcfosshelplinein/smedia/images /downloads/Tamil Tagset-opensourceodt.

121 Vishal Goyal, Suman Preet,Navneet Garg. *Rule Based Hindi Part of Speech Tagger*. Proc. of the COLING, pp. 163–174, Mumbai, December 2012.

122 Viterbi A J. *Error bounds for convolutional codes and an asymptotically optimal decoding algoritm*. IEEE Transaction on Information Theory, Vol.13:pp. 260-269, 1967.

123 Wallach H M. *Efficient training of conditional random fields Master's thesis*. University of Edinburgh, 2002.

124 Yamamoto, Y and Sakamoto.M. *Extraction of Technical Terms Bilingual Dictionary from Bilingual Corpus*. IPSJSIG Notes Natural Language 094-012, 1993.

125 Zipf. *Selective Studies and the Principle of Relative Frequency in Language Harvard University Press*. Cambridge, MA, 1932.

Web Resources

126 http://crfppsourceforgenet/

127 http://enwikipediaorg/wiki/Named_entity_recognition (Accessed on 15th SEP 2009)

128 http://ltrciiitacin/ner-ssea-08/indexcgi?topic=3

129 http://nlpersblogspotcom/2006/08/doing-named-entity-recognition-donthtml

130 IIIT-tagset, A Part-of-Speech Tagset for Indianhttp:// shivaiiitacin/SPSAL2007 /iiit_tagset_guidelines Pdf

131 URL= http://ltrciiitacin /online services/ Dictionaries/Dict_Framehtml

132 AU-KBC.*POS Tagset for Tamil http://nrcfosshelplinein/ smedia/ images/ downloads /Tamil.*

133 http://www.cfilt.iitb.ac.in/wordnet/webhwn/

ABSTRACT

Natural language processing (NLP) is a branch of artificial intelligence (AI) that studies the problems of automated generation and understanding of natural human languages. The goal of NLP is to design and build software that will analyze, understand, and generate languages that humans use naturally. Natural Language Processing (NLP) is a convenient description for all attempts to use computers to process natural language. Natural Language Processing (NLP) includes Natural Language Understanding (NLU) and Natural Language Generation (NLG). This thesis is about Automatic Morphosyntactic Annotation System (AMAS) for Kannada language using hierarchical POS tag set. Annotation for Kannada is challenging since Kannada is a highly inflectional and agglutinative language. Kannada language provides richest and challenging set of linguistic and statistical features. There are few languages in the world that match Kannada in this regard. There is not much work done in annotation or tagging for Kannada language.

Kannada is a language of Dravidian family which is spoken mainly in the southern part of India and ranks third among Indian languages in terms of number of speakers. Kannada is a highly inflectional and agglutinating language, providing one of the richest and challenging set of linguistic features. There are very few languages in the world that match Kannada in this regard. There is not much work done in Annotation in Kannada language. Annotation system developed here is not merely Part of Speech (POS) tagger, it performs annotation at 3 levels dictionary tagging, morphological tagging and named entity recognition (NER) tagging. Developing all these resources for Kannada is not an easy job. Kannada is challenging as it is very rich in morphology, giving rise to a very large number of word forms.

Preliminary studies show that the existing dictionaries and morphological analyzers are incomplete and imperfect and need substantial improvements to develop a good Annotation System. Annotation is the process of adding some additional information (grammatical features like word category, case indicator, other morph features) to each word in the text. This additional information is called a tag. The set of all tags is called a tagset.

The research work proposed an automatic morpho-syntactic annotator system (AMAS) for Kannada language, using hierarchical part of speech (POS) tagset. The annotator system includes design and development of five modules viz, Design and development of hierarchical tagset, Development of an exhaustive morphological analyzer/generator, Design and development of an electronic dictionary, design and development of named entity recognizer (NER) module using rule based and machine learning approaches. **The main aim of this work is to achieve functional automated annotation system for Kannada**.

Hierarchical Part of Speech (HPOS) tag set is proposed at the first stage of the research work*. The tag set has many** advantages as compared existing flat tag sets (IIIT-Hyderabad), (AUKBC) tagsets in terms of capturing deep morpho-syntactic information than the flat tag set. ***The proposed HPOS has overcome many open issues left unsolved in (Baskran et al.) IL-POST tag set. The HPOS tag set forms the basis for many Natural Language Processing (NLP) applications like parsing and Machine Translation (MT), where detailed information of the word is useful. ***A total of 171 atomic elements are used in building the tag set***. European Advisory Group on Language Engineering Standards (EAGLES) is used as model with language specific modifications to capture the inherent variability of complex Kannada morphology. Designing and developing hierarchical tag set for Kannada like languages is ***challenging.***

An electronic dictionary using semi-automatic and Heuristics methods is developed in second stage of research work. Each word in the dictionary is tagged with hierarchical tag set. Developing a dictionary using hierarchical tagset is new attempt as far as Kannada is considered. Developing dictionary with flat tag set is just mentioning the category of the word simply as noun or verb etc., but designing a dictionary with hierarchical tag set is again tedious effort full task, because every morpho-syntacic information like countable, uncountable, common, singular, gender related specifications have to be marked on each entry. Marking of uncountable information on nouns is important since it avoids morphological generation process

to become over general, which otherwise generates un-syntactic word forms. ***We have compiled a dictionary of around 30,000 words.***

The work proposed an Exhaustive Morphological Analyzer and Generator (MAG) system using finite state transducers (FST). The morph system uses hierarchical POS tag set and dictionary. ***Developing Morphological analyzer/generator using hierarchical tag set is also a new attempt as far as*** **Kannada Language is considered. The already existing morphological analyzers/generators are built using flat tagset.** The proposed MAG system handles **inflectional morphology, derivational morphology** and **External SaMdhi and also dialectic variation**. The MAG system proposed here has very high performance as compared to already existing morphological systems for Kannada (Shalini urs, 2008), (Shambhavi, 2011), (K N Murthy, 1999). Developing morphological analyzer and generator for languages like Kannada is challenging due to the inherent complex word formation process. The complexity of word formation process is comparable to world's complex morphology languages like Finnish and Turkish. The results regarding annotation using morphological analyzer are encouraging in our work and around 90% for nouns and 81% for verbs.

The research work proposed a rule based methodology to recognize Kannada named entities like person name, location name, organization name, number, measurement, time. We have manually developed suffix, prefix list and proper noun dictionary of **5,000** words. Lack of capitalization feature in Kannada, lack of gazetteers makes **Kannada NER challenging**, proper nouns are indistinguishable from forms that are acting as common nouns and adjectives for example city name "Bangalore" is location name which is noun, and also surname of person. This kind of ambiguity makes Kannada NER more challenging. The results of recognition are encouraging and the methodology has the precision and recall around 86% and 87% respectively. Famous Kannada news paper Prajaavaani corpus is used to carry out experiments. ***Rule based attempt for NER is first attempt for Kannada Language. There is no hard coding in the programming and approach is language independent and can be used for other languages just by replacing the language related stuff.***

A Statistical methodology using HMM is also attempted for Kannada NER Task. Manually tagged data of 2100 words is used for training. The method gives accuracy is around 93%. HMM is implemented using Java, JDK 1.2 NET BEANSIDE, in Windows. Input to HMM is pdf, docx, doc and txt files. The HMM transitions are passed to Graphviz transition diagram drawer to drqw the transition diagram which indicates the transitions from one state to another. This is an added feature for existing HMM.

Another Machine learning technique using Conditional Random Field (CRF) is ***also developed to recognize Kannada named entities***. Conditional Random Fields (CRFs) are a probabilistic framework for labeling sequential data based on the conditional approach. The primary advantage of the CRF over the HMMs is the conditional nature, resulting in the relaxation of the independence assumption required by HMMs. We have achieved higher accuracy in CRF approach around 96% than the HMM model. The accuracy of classification is more in CRF approach due to inclusion of long dependency and addition of more features rather than joint probability alone as in HMM.

In short the work carried out comprises of development of various lexical resources for tagging of Kannada words. The resources include design and development of hierarchical tag set, morph related dictionary, morphological analyzer and Generator and named entity recognizer. These resources can be used as components of Machine translation (MT) systems, or in chunking applications, in developing tagged data. Tagged data is useful to carry out statistical experiments. Our morphological analyzer can be used as language learning tool since the output of the analyzer clearly depicts word formation process. Proposed annotator is an exhaustive system.

The significance of large annotated corpora in the present day Natural Language Processing (NLP) is widely known. Annotated corpora serve as an important tool for investigators of natural language processing, speech recognition and other related areas. It proves to be the basic building block for constructing statistical models for automatic processing of natural languages. We have proposed a novel architecture for a wide coverage automatic morphosyntacic annotator system for Kannada language.

Contents

LIST OF BOXES

LIST OF FIGURES

LIST OF TABLES