Statistical POS Tagger for Sanskrit: Methods, Modality & Challenges

Thesis submitted to Jawaharlal Nehru University

In partial fulfillment of the requirements

For the award of the

Degree of

DOCTOR OF PHILOSOPHY

Archana Tiwari

Special Centre for Sanskrit Studies

Jawaharlal Nehru University

New Delhi-110067

2016

Statistical POS Tagger for Sanskrit: Methods, Modality & Challenges

Thesis submitted to Jawaharlal Nehru University

In partial fulfillment of the requirements

For the award of the

Degree of

DOCTOR OF PHILOSOPHY

Archana Tiwari

Special Centre for Sanskrit Studies

Jawaharlal Nehru University

New Delhi-110067

2016

विशिष्ट संस्कृत अध्ययन केन्द्र
जवाहरलाल नेहरू विश्वविद्यालय
नई दिल्ली-११००६७

Special Centre for Sanskrit Studies
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067

Date: 21 July, 2016

DECLARATION BY THE CANDIDATE

This Thesis titled "**Statistical POS Tagger for Sanskrit: Methods, Modality & Challenges**" submitted by me for award of the degree of Doctor of Philosophy, is an original work and has not been submitted so far in part or in full, for any other degree or diploma of any University or Institution.

Archana
(Archana Tiwari)

विशिष्ट संस्कृत अध्ययन केन्द्र
जवाहरलाल नेहरू विश्वविद्यालय
नई दिल्ली–११००६७

Special Centre for Sanskrit Studies
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067

July, 21, 2016

CERTIFICATE

This Thesis titled "**Statistical POS Tagger for Sanskrit: Methods, Modality & Challenges**" submitted by **Archana Tiwari** to **Special Centre for Sanskrit studies, Jawaharlal Nehru University, New Delhi-110067**, for the award of the degree of **Doctor of Philosophy**, is an original work and has not been submitted so far, in part or full, for any other degree or diploma of any University.

This may be placed before the examiners for evaluation and for award of the degree of **Doctor of Philosophy**.

[illegible]

SUPERVISOR
Dr. Girish Nath Jha
Professor
Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi - 110067

Dr. Girish Nath Jha
[Supervisor]

[illegible]

PROF. GIRISH NATH JHA
Chairperson
Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi-110067, INDIA

Dr. Girish Nath Jha
[Chairperson]

Dedicated to Nana Ji

and

Ma - Papa

Acknowledgement

I acknowledge with a deep sense of gratitude, continuous help and encouragement from my supervisor Dr. Girish Nath Jha, which he has provided to me despite his busy schedule. Without his able direction, active supervision, positive attitude and motivation, it would have been extremely difficult for the present research and development to complete. In the able hands of Dr. Jha, I have been inducted into the vast arena of computational linguistics (CL), Natural Language Processing (NLP) and Artificial Intelligence (AI). He helped me from time to time in programming, RDBMS techniques, text files and dissertation writing and solved several difficulties. He has been instrumental in my induction into more specific fields such as parts of speech tagging, statistical modelling for Indian languages, computational morpho-syntax and syntax. I have learnt from him how to create statistical models understanding the structures of the languages. Without him this novel undertaking could never be accomplished.

I express deepest thanks and gratitude for Chairperson, Prof. Shashiprabha Kumar and other faculty members - Dr. Hariram Mishra, Dr. Rajnish Kumar Mishra, Dr. Santosh Kumar Shukla and Dr. Ram Nath Jha - of the ***Special Center for Sanskrit Studies (SCSS), Jawaharlal Nehru University, New Delhi****, for encouraging and allowing this research to complete.*

I express my special thanks to ***Pitambar*** *and* ***Shristi*** *for helping me time to time. My special thanks to my junior* ***Pragya*** *for helping me in annotation work. Special thanks are also due to* ***Atul Kumar Ojha*** *who has assisted me linguistically and computationally and configured the statistical models employed in this research.*

I also express my sincere thanks to all the office staff of my centre, and those who directly or indirectly helped me for completing this work.

I am really thankful to my classmates with whom I have learned many things and enjoyed the learning process. I specially want to thank Rajanish. He helped me in during last chorusof thesis printing and submission.

My warmest thanks to ***Kaushal****, for his continued and unfailing love, support and understanding underpins my persistence in the compellation of this thesis. Thank you for your support and love. I could never have done it timely if I hadn't had you there with me.*

I want to express my gratitude to my friends ***Roma, Sudha, Pranjal , Shraddha and Pooja****. They have always been with me in hot or cold days.This journey could not have been completed without you girls.*

With deep sense of indebtedness, I express my heartfelt gratitude to ***Mummy-Papa, Ma- Pitaji*** *for their affection and immense support. I warmly thanks to my Sisters,* ***Richa, Puja*** *and brother* ***Vaibhav*** *for their constant motivation and invaluable sacrifice all through my studies. They have been guiding light at every crucial juncture.*

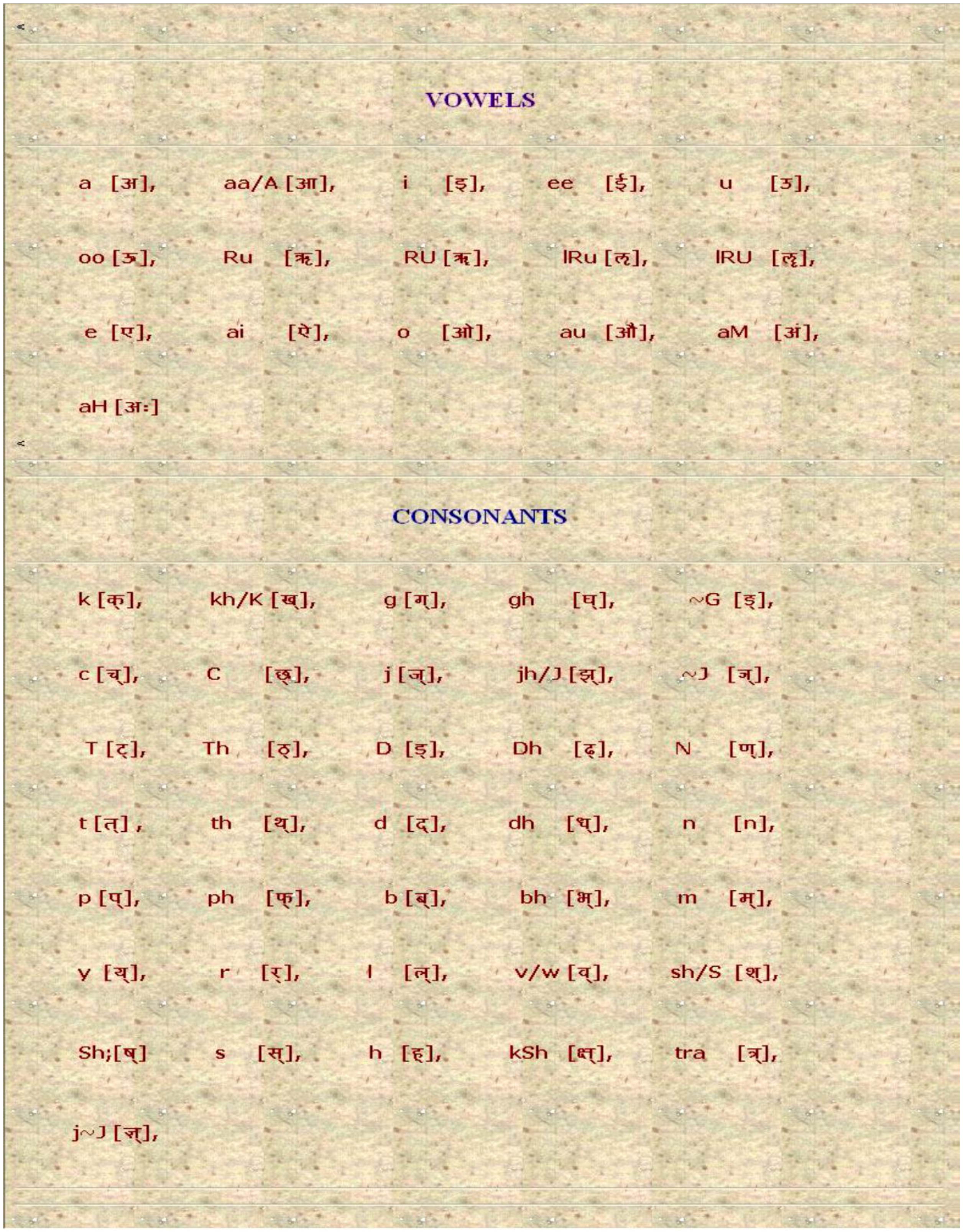
I also express my sincere thanks to ***Atama Prakash*** *and all the office staff (Shabanam Mam, Manju Mam,Arun ji and Vikas) of my centre, and those who directly or indirectly helped me for completing this work.*

Archana Tiwari

Contents

Transliteration key used in the Thesis

अ	=	a	ण्	=	*ṇ*
आ	=	*ā*	त्	=	t
इ	=	i	थ्	=	th
ई	=	*ī*	द्	=	d
उ	=	u	ध्	=	dh
ऊ	=	*ū*	न्	=	n
ऋ	=	*ṛ*	प्	=	p
ॠ	=	*ṝ*	फ्	=	ph
ऌ	=	*ḷ*	ब्	=	b
ए	=	e	भ्	=	bh
ऐ	=	ai	म्	=	m
ओ	=	o	य्	=	y
औ	=	au	र्	=	r
क्	=	k	ल्	=	l
ख्	=	kh	व्	=	v
ग्	=	g	श्	=	*ś*
घ्	=	gh	ष्	=	*ṣ*
ङ्	=	*ṅ*	स्	=	s
च्	=	c	ह्	=	h
छ्	=	ch	क्ष्	=	k*ṣ*
ज्	=	j	त्र्	=	tr
झ्	=	jh	ज्ञ्	=	j*ñ*
ञ्	=	*ñ*	ऽ	=	’
ट्	=	*ṭ*	ं (*Anusvāra*)	=	*ṃ*
ठ्	=	*ṭ*h	ः (*visarga*)	=	*ḥ*
ड्	=	*ḍ*			
ढ्	=	*ḍ*h			



VOWELS

a [अ],	aa/A [आ],	i [इ],	ee [ई],	u [उ],
oo [ऊ],	Ru [ऋ],	RU [ॠ],	lRu [ऌ],	lRU [ॡ],
e [ए],	ai [ऐ],	o [ओ],	au [औ],	aM [अं],
aH [अः]				

CONSONANTS

k [क्],	kh/K [ख्],	g [ग्],	gh [घ्],	~G [ङ्],
c [च्],	C [छ्],	j [ज्],	jh/J [झ्],	~J [ञ्],
T [ट्],	Th [ठ्],	D [ड्],	Dh [ढ्],	N [ण्],
t [त्] ,	th [थ्],	d [द्],	dh [ध्],	n [n],
p [प्],	ph [फ्],	b [ब्],	bh [भ्],	m [म्],
y [य्],	r [र्],	l [ल्],	v/w [व्],	sh/S [श्],
Sh;[ष्]	s [स्],	h [ह्],	kSh [क्ष्],	tra [त्र्],
j~J [ज्ञ्],				

Devānagarī Input Mechanism by using Baraha Software

List of Abbreviations Used

AD	Aṣṭādhyāyī
AI	Artificial Intelligence
BIS:	Bureau of Indian Standardization
BNC:	British National Corpus
CLAWS	Constituent Likelihood Automatic Word Tagging System
CPOS	Sanskrit Consortium Tagset
CRF	Conditional Random Fields
DeitY	Department of Electronics and Information Technology
GP	Ga*ṇ*ap*āṭ*ha
HMM	Hidden Markov Models
IA	Indo-Aryan
ILMT	Indian Language Macune Translation
IL-POST	Indian Language – Part of Speech Tagset
IR	Information retrieval
JPOS	JNU-Sanskrit Tagset
LDC-IL	Linguistic Data Consortium for Indian Languages
LOB	Lancaster-Oslo/Bergen
MBL	Memory based learning
ME	Maximum Entropy
MET	Maximum entropy Models
ML	Machine learning
MorphA	morphological analyzer
NL	Natural language
NLP	Natural Language Processing
POS	Part Of Speech
POST	POS Tagger
PS	Praty*ā*h*ā*ra S*ū*tra
SLR	Structured linguistic Resource
SP	S*ū*trap*āṭ*ha

SVM	Support Vector Machine
TBL	Transformation-Based Learning
TDIL	Technology Development for Indian Languages
TDIL	Technical Development of Indian Languages
TNT	Trigrams N Tags
TTS	Text to Speech
UIMA	Apache Unstructured Information Management Architecture

List of BIS Abbreviations Parts of Speech Tags

CC_CCD	Coordinating Conjunction
CC_CCS	Subordinating Conjunction
DM_DMD	Deictic Demonstrative
DM_DMI	Indefinite Demonstrative
DM_DMQ	Interrogative/wh Demonstrative
DM_DMR	Relative Demonstrative
JJ	Adjective
N_NN	Common Noun
N_NNP	Proper Noun
N_NST	Spatial and Temporal Noun
N_NNV	Verbal Noun
PSP	Postposition
PR_PRC	Reciprocal Pronoun
PR_PRF	Reflexive Pronoun
PR_PRI	Indefinite Pronoun
PR_PRL	Relative Pronoun
PR_PRP	Personal Pronoun
PR_PRQ	Interrogative/whPronoun
QT_QTC	Cardinal Quantifier
QT_QTF	General Quantifier
QT_QTO	Ordinal Quantifier
RB	Adverb
RD_ECH	Reduplicative Echo Word
RD_PUNC	Punctuation
RD_RDF	Foreign Word
RD_UNK	Unknown Word
RD_SYM	Symbol/Special Character
RP_CL	Classifier

RP_NEG	Negation
RP_INJ	Interjection
RP_INTF	Intensifier
RP_RPD	Default Particle
V_VAUX	Auxiliary Verb
V_VM	Main Verb
V_VM_VF	Finite Verb
V_VM_VINF	Infinitive Verb
V_VM_VNF	Non-Finite Verb
V_VM_VNG	Gerundive Verb

List of Tables, Figures, Screenshots, Charts

INTRODUCTION

India is a country having diversity of languages. In the huge verity of these languages Indo-Aryan Languages and Dravidian Languages are major linguistic families. A vast amount of Natural Language Processing (NLP) work has been done using Corpus based techniques for popular languages like English, Greek etc. These techniques have been worked with a great success. On the contrary, only small amount of NLP task has been done on Indian languages. There are quite a few reasons for this. First of all the main reason is unavailability of annotated ready to use structured linguistic Resource (SLR) for such languages

The main reason behind less corpus based NLP work for Indian languages is that there is a sacristy of annotated corpus. Lack of ready to use corpus source for Indian languages make the NLP task difficult.

The second reason which makes NLP work difficult is that, Indian languages are morphologically rich and agglutinative in nature. Because of these feature makes the task of creating efficient language specific tool is hard to achieve.

Part Of Speech (POS) tagging is the solution for these problem faced by Indian language NLP researcher.

"POS tagging may be defined as the process of assigning to each word in a running text a label which indicates the status of that word within some system of categorizing the words of that language according to their morphological and/or syntactic properties".[1]

The tool that does POS tagging is called a tagger. The work of a tagger is to assigns a (unique or ambiguous) POS tag to each and every token in the input given by the user and generates its output to the next level of processing.

POS is also known as word-classes, morphological classes or lexical categories in a language. The reason why POS is so important for the languages processing is that it not only gives significant amount of information about that word, but about the features of its neighbouring words.

POS tagging can be used in TTS (Text to Speech), information retrieval, shallow parsing, and information extraction, linguistic research for corpora and also as an intermediate step for

[1] Hardie, Andrew (2003). "*The Computational Analysis of Morpho-Syntactic Categories in Urdu*", *Ph.D. Thesis* submitted to the Dept. of Linguistics and Modern English, Lancaster University, (revised soft copy 2004), p. 40.

higher level NLP tasks such as parsing, semantics, translation, and many more. POS tagging, thus, is a necessary application for advanced NLP applications for any language.

A POS Tagger (POST) is an kind of software that take, text in some language as input and marks parts of speech to each word or token, such as noun, pronoun, verb, adjective, punctuation etc.

A general classification of tagger is :-

1) Rule-based taggers.

2) Statistical Tagger

3) Hybrid taggers

In rule based POS tagging models, a set of hand written rules is used which is applied on an input text. Then it uses syntactical and contextual information of the word for assigning POS tags to them. Those rules are used to distinguish the tag ambiguity. The two stage architecture is used in developing these kind of taggers. First stage is a dictionary and in the second stage hand written rules are stored. “Brill’s tagger” was the very first English POS tagger, which employ this rule based algorithm.

The empirical based taggers are corpus based. There are two techniques for development of taggers in this approach, example based and stochastic. They use a machine to learn and extract rules from a particular languages corpus and assign those rules in the unannotated text.

In the hybrid approach the tagger uses both hand-written and probabilities methods. It learns coarse information of language from the corpus through empirical technique. For minute and specific problems hand written rules are used.

Scope & objectives

Sanskrit is a classical language of India. It has the biggest collection of literature. Literature from every part of human life has found its place in this vast collection. Almost all branches of knowledge are preserved here, whether it is technology, medicine mathematics, astronomy or literature, medicine, logic, dramatics, poetics. These names are very few from the vast area of Sanskrit.

Till the recent times, all the scholarly communications and serious discussions were communicated through Sanskrit, it resulted in a continuous production of literature in Sanskrit.

As quoted by Kulkarni et.al. (2012),"Sanskrit in various branches of knowledge systems of human endeavour has been done for almost over 6000 years. The corpus of Sanskrit is 100 times more than Greek and Latin language corpus put together"[2]. But in the last two centuries the whole scenario has undergone drastic change. During colonial times, the way of traditional learning methods were got replaced by the western learning system. This change made the knowledge of Sanskrit text inaccessible, not only for common masses but for modern Indian scholars also."[3]

Emergence of computational linguistics, has given a new opportunity. Now it is possible to build tools for accessing and processing of Sanskrit. There is a problem in this path, as Computational Linguistics is developed in western world, so it is based on the theory and structures which are suitable for English and other western languages.

On the other hand, Sanskrit is a language of rich morphological features. The study of it, is dominated by oral tradition. Due to being influenced by an oral tradition Sanskrit has a lot of variation. Sanskrit texts have long threads of characters which lack word and sentence boundaries and punctuation marks. It not only does merging of the word boundaries but sometimes the sentence boundaries as well. A very rich and inflectional morphology makes Sanskrit processing a difficult task. In Sanskrit, it is quite difficult to remember various inflections of a word, which most of the time differ with the last characters of the words or the gender of the word. Therefore, this presents immense challenge in the computational processing of Sanskrit texts. This is the reason, why pre-processing of Sanskrit text is crucial for obtaining the knowledge preserved in them.

In present research, the task of building a statistical Part-of-Speech (POS) tagger for Sanskrit has been done. There are several machine learning approaches, but I have used the Support Vector Machine method for training tagger model. It will be useful for the research and Development of the 'Sanskrit-Indian Languages Machine Translation Systems (MTS)'. For a

[2] Kulkarni Amba and Monali Das. (2012). *"Discourse Analysis of Sanskrit text"*, Proceedings of the Workshop on Advances in Discourse Analysis and its Computational Aspects (ADACA). COLING 2012, Mumbai, December 2012, pages 1–16
[3] Ibid.

given *sandhi*-free classical Sanskrit prose text, the proposed system will assign correct POS tag for each word through the statistical approach of POS tagging.

The research aims at two main objectives -

1. To develop a statistical POS tagger for Sanskrit by using machine learning approaches.
2. Study of the issues emerging in automatic tagging of Sanskrit texts by machine.

The POS tagger, prepared in this work, is first of its kind for Sanskrit language. Several Sanskrit rule-based taggers have been developed. A Sanskrit POS tagger developed by Oliver Hellwig, is a stand-alone and take input of un-pre-processed data. One tagger has been developed by R. Muni et.al of Tiripati using treetagger.

But POS tagger developed during this research work would take pre-tagged 155k tokens from different texts as input. In this work data would be tagged with a National standard tagset, BIS. The tagger would be an online one.

Rationale for Statistical POS Tagger for Sanskrit

The formalized structure of Sanskrit Language is well known for the usefulness of being a rich source for NLP related activities. But there is a major obstacle for getting access to the knowledge of this resource. In Sanskrit; intense saṃdhi and samāsa (compound) formation is the major challenge. Even Scholars trained in modern linguistics have difficulty in

Sanskrit texts because of their intense saṃdhi and samāsa (compounds) formations are very difficult to access for a person trained in modern languages. Therefore, to facilitate a modern reader to access texts in Sanskrit, it is necessary to annotate such texts based on traditional principles of language analysis.

POS information is very important for language processing because it gives significant amount of information about the word and its neighbours. This is clearly true for the major categories, like quantitative *sarava* versus pronoun *sarva* : the quantitative *sarva* expects a noun in but a Pronoun *Sarva* is used instead of a noun.

For other finer distinctions, the case is same. A word's category can be distinguished by the case-inflection sub-tags and gender-number sub-tags. For example; *bālakāḥ gacchanti* in this sentence, the number tag in the verb '*gacchanti'* demands the nominative singular of the word '*Balaka', that is 'Balakaḥ'.* POS information of a word can tell which word with which POS is more likely is going to occur in its vanicty.

Thus tagging can be useful in making information retrieval system, language model for speech recognition etc. A word's POS can give information about how the word is pronounced. The word *api*, for example, can be a interrogative particle or a separable adverb or conjunction. The presence of the type of *api* will have difference in the prosody of the sentence (for example in the sentence: *api rāmaḥ gacchati*? Interrogative *api* will have interrogative prosody in the sentence). Thus the POS mark can generate more natural and accurate pronunciations in a speech synthesis system and by this it can enhance the accuracy of the speech recognition system.

The complexity of ambiguity in Sanskrit can be shown using an example word '*bhavati*'. It can be a verb or pronoun or a participle. It can still have many more ambiguous forms within the above mentioned categories, if we take inflectional features also into consideration for tagging. The few forms of the present participle forms of the √*bhū* in its three genders can be similar to the second person pronominal '*bhavat'* in its three genders. And also the present tense third person singular of the √*bhū*.

durgasthaḥ duḥsādhyaḥ ***bhavati***(Verb Present First Person Sing.) *ripuḥ*.[4]

dr̥ṣṭe ***bhavati***(Pronoun Mas. Loc. Sing.) *pra-****bhavati*** *na* ***bhavati***(verb present 1.1)

kiṃ bhavatiraskāraḥ.[5]

bhavati(Pronoun Fem. Voc. Sing.) *bhiksāṃ dehi*.[6]

Though, the use of *bhavati* in the sense of pronoun in neuter gender and as a participle is not a common phenomena, still grammatically it cannot be ruled out.Thus word class ambiguity resolution is really a serious problem in Sanskrit.

---

[4] Pancatantra, Mitrabheda Katha 8

[5] Viṣṇuṣaṭpadīstotra 4

[6] Saundaryalaharī-Bhāṣya of Lakṣmīdhara

Approach/Method/Technique

The methodology used in present research and development of the tagger is, Sanskrit based computational linguistics, machine learning and software engineering will be used.

The system will adopt the statistical learning approach. In the statistical learning approach, the system will learn from annotated corpus and apply it on unseen text. The stochastic (probabilistic) approach learns from a training corpus to pick the most probable tag for a word.

The methodology can be elaborated as:

1. **Collecting Sanskrit data**

For development of a statistical tagger one should have a vast amount of data. As from that data the tagger learns the language features. In this work the data has been collected from different Sources and domains. Sanskrit texts like *Panchtantra*, *Hitopadeśa* are the main source for collection of data from literature domains. Some other blogs e.g. Balaram Sukla's blog[7] is used for data collection. *Sudharma* news paper provided the data from general domain. 100k sentences has been collected for tagging and training of the tagger.

2. **Checking the standard of the data and normalising it**—

Long sentences with complex structure (*sandhi-samāsa*) are frequent in Sanskrit . For training the machine, standardization of these sentences is required. Using sandhi-splitter, pause and semicolon the normalization of the text would be done. It would make the data more understandable for machine.

3. **Annotation of the data**

For training the tagger data annotation is the primary task. For annotation of the data a standard tagset developed by bureau of Indian standard has been used. The BIS POS Tagset is a national standard tagset developed for Indian languages. It has been designed recently by the POS standards committee at IIIT, Hyderabad. It is hierarchical tagset, without morpho-syntactic features. This tagset has 11 categories at the top level. Conversion of the tagset is the primary task of this work. First 77,000 token has been taken from literature domain for

[7] https://balramshukla.wordpress.com/

training. 78,000 token has been collected from different domains from Sudharma Newspaper and other Sanskrit blogs.

4. Selection of computing algorithm

Selecting an appropriate computing algorithm for Machine learning and developing a statistical POS tagger is a major task. The well known and widely used machine learning techniques for POS tagging are-

a. Hidden Markov Models (HMM)

b. Conditional Random Fields (CRF)

c. Maximum entropy Models (MET)

d. Support Vector Machine etc. (SVM)

In this work the researcher has used Support Vector Machine for developing the tagger. Morphological richness of Sanskrit requires such a tool which is flexible enough to handle the morphological complexities of the language. So far, SVM has been proved to be an efficient device with its variant, for resolving different kinds of issues like quadratic programming, optimization problem, problem of dual form, soft margin approach for the problem of mislabelled examples, and many more.

5. Training the POS Tagger

For this tagger the machine has been trained by the training sets consisting 155k tokens. The machine learning system will be provided example sentences **(training set)** in which words are marked with their appropriate POS tags. The machine can encode the learnt information using a set of parameters or rules or both. The system will then use this learnt information to tag new sentences **(testing set)**.

6. Testing the result

After training, the results will be tested. The issues where the machine will have problems in tagging the word with its correct tag will be analyzed.

7. **Analysis of the errors**

In this section the tagger will be analyzed that How far it is able to handles the problems arise during tagging.

Chapters

- Introduction
- Chapter 1: POS Tagging Concepts and Practice

  This chapter gives a description about theoretical aspect of POS tagging. Definition, concepts and categorisation of POS has been given in detail. POS tagger and requisites for it e.g. corpora, tagset etc has been described. Machine learning, its importance in NLP and different methods in Machine learning is discussed in the second part.

- Chapter 2: Survey of POS taggers

  This chapter has two parts. In the first part the chapter has a survey of taggers prepared for in world wide. In the second part description of the taggers in indian languages has been a topic of discussion. Taggers and their architecture and results are the points of discussion in this chapter.

- Chapter 3: POS Tagging in Sanskrit Language

  The history of POS in Sanskrit is the initial discussion topic in this chapter then Sanskrit morphology has been elaborated fruther. Then problems during tagging of Sanskrit text were discussed. Second part contains a detailed description of Sanskrit taggers and tagging systems

- Chapter 4 : Sanskrit Tagset and Preparation of the Tagged Data

  This chapter contains information about Sanskrit morphology and tagset which has been used for annotation. A detail description of BIS tagset, which has been used in preparing tagged data, will be discussed in the second part of the chapter. Then a sample of tagged data is given in the end.

- Chapter 5 : Training the POS Tagger,Testing and Evaluation

  This chapter is technical in nature, as it gives the detail of the tagger's system. The testing methods and results after training is the second part of the chapter. The problem in tagged data from system is also discussed.

- Conclusion and Future R&D

  This part will give a point of view for future research and development in the area of POS tagger development through use of Statistical methods.

Chapter 1:

POS Tagging Concepts and Practice

Introduction

This chapter deals with the theoretical part of this work. The first part of the chapter deals gives basic information about the Natural language processing. Part of speech and POS tagging, it classification and different types are other topics of discussion. The second part of the chapter discuses about machine learning and its usefulness in the language technology. This part gives a brief description of different approaches of machine learning.

1.1. Natural language Processing (NLP)

Natural language processing (NLP) is an area of computer science, artificial intelligence, and linguistics. NLP area of concern is the interactions between computers and human (natural) languages. NLP tries to develop a program which can understand human language the way it is spoken. It is a sub-field of computer science, linguistics and artificial intelligence.

The objective of NLP is to comprehend the structure and function of human language with a better insight and with this construct a better natural language interface. By this, making communication between humans and computer can become a reality. NLP tries to develop a system that can read text and translate from one human language to another. In this way the first and basic system is human-computer interaction. With the help of this system, NLP tries to make human able to communicate with computer using their everyday languages. Other prominent uses of NLP are Information retrieval, Speech generation and speech processing.

Natural language understanding is required to handle various challanges in NLP. By natural language understanding means enabling computers to derive meaning from input given in human or natural language. To build NLP application is a difficult because human speech is not always specific. One other major challenge is natural language generation.

For development of NLP applications the initial challenge faced by the developer is shallow linguistic information, in which includes part–of–speech tagging, base phrase chunking, named entity recognition.

For these challenges, Part Of Speech (POS) tagging is the first step of the solution. POS tagging is the first, basic and essential part of natural language understanding and generation. For any NLP task a well annotated (tagged) corpora is the pre-requisite.

In the beginning corpora annotation had been done manually, but it was very time-consuming, labour-intensive and error-prone. The high level of ambiguity was also a major concern. That's made the automated tagging a need of the hour. This requirement makes corpora annotation a burning topic. The reason is, its importance as the basis of research and development in language technology and computational linguistics.

Many NLP applications require POS tagging as their initial stage as it acts like a shallow parser. A POS tagger is the pre requisite tool for any NLP system be it, machine Translation tools, Word sense disambiguation tools, Speech synthesis and Speech generation tools, Information retrieval (IR) tools or Spell checkers etc. It is also the primary step in the syntactic analysis of a language.

1.2. Word Classes and Part-of-speech

Words are divided into different categories. These categories can be named as parts of speech (POS; Latin pars orations), word classes, morphological classes, or lexical tags. There are several parts of speech in the traditional grammars (noun, verb, adjective, preposition, adverb, conjunction, etc.).

According to MacKinlay (2005), "Modern linguistic approaches often use the term word class rather than part of speech to avoid a term with too many pre-existing associations. Recognizing the shortcomings of the semantic definitions of POSs, they tend to identify word classes by formal criteria. One such criterion is based on syntax: word classes should be comprised of words which tend to occur in a particular position relative to other words in a sentence"[8]

Classification of POS

POS is usual categorized in two broad categories of word classes: open classes and closed classes.

Open class- Open classes can be recognised by these characteristics: One of them is that these class freely include new words. The other feature of open class is that the words

[8] MacKinlay Andrew (2005). *"The Effects of Part-of-Speech Tagsets on Tagger Performance". Ph.D.Thesis* Submitted to The Department of Computer Science and Software Engineering ,University of Melbourne.

contained in them easily get modified by morphological process. These words generally make up a huge percentage of the lexicon of the language.

Closed class- A very small number of entries are included in this class. But in a running text these words make a very large percentage of the tokens. This category does not readily admit new words. This class contain words which allow only a little or no morphological modification. The example of closed class words are, Conjunction, preposition etc. Present time the models have a large number of word classes to handle the ambiguity of the language.

Penn Treebank has 45 word classes. The Brown corpus has 87 and C7 tagset has 146 word classes.

1.3. What is POS tagging?

POS tagging is a process to assign part-of-speech tags in a corpus based on the information o that word's definition and context. In a text, a part-of-speech refers to a grammatical category. This grammatical category includes verbs, nouns, adjectives, adverbs, determiner etc. POS tags are also known as word classes, morphological classes or lexical tags. POS tagging is a classification task of NLP, in which the input is an unlabelled of text (or corpus).The output of it is a word token assigned with a POS. Here term "POS" is more similar to what we described as "word classes" above than the POS from traditional grammars.

According to Hardie, "POS tagging may be defined as the process of assigning to each word in a running text a label which indicates the status of that word within some system of categorizing the words of that language according to their morphological and/or syntactic properties".[9]

In the pos tagging the Input has a string of words and a tagset consisting tags for POS categories. The output is the single best tag for a word which is suitable according to that words property.

POS tags give a lot of information about the words and its neighbouring words. The basic function of these tags is to indicate syntactic categories of the word, such as noun or verb.

[9] Hardie, Andrew (2003). The Computational Analysis of morpho-syntactic categories in Urdu. *Ph.D. Thesis* submitted to the Dept. of Linguistics and Modern English, Lancaster University, (revised soft copy 2004), p. 40.

Some time, they also give information about additional features, e.g. number (the word is singular or plural) gender and verb tense.

The tagging process can be handled on two levels-

1. Word-level
2. Sentence-level.

At word-level POS tagging is posed as a classification problem in which an appropriate tag for a word is found. At the sentence level a series of tags corresponding to the sequence of words are obtained.POS tagging resolve the issue of ambiguity of words in a sentence or corpus. It can be manually or automated.

Basically, tagging is one of the prototypical problems faced in resolving lexical ambiguity. If the in part-of-speech tagging could be advanced then it easily translate to progress in other areas of lexical and structural ambiguity solving. This will make, word sense disambiguation and prepositional phrase attachment disambiguation task convenient. Letter-to-sound translation and constructing pronunciation network for speech recognition is one of the problems in which POS tagging contribute.

1.3.1 Why POS tagging

This question can be answered in these points:-

1. Pos tagging is a simple task which has usually linear processing time so It can be used in many other applications like-Text-to-speech, Speech recognition, parsing, information retrieval, etc.
2. More abstract levels of analysis benefit from reliable low-level information; so pos tagger can work as a pre-processor for a parser; it makes the parser's work easy and speeds it up. Although parser can does it better but parser is more expensive and complex?
3. Information technology applications like text indexing and retrieval can benefits from POS information. Speech processing tools an also benefits from tagging.

4. One of the advantages of the POS tagging is that it is easy to evaluate. It easy to analyze that how many tags are correct?

5. In linguistic studies large tagged corpora could be really useful for analyzing any language.

6. Data size should be sufficient for training. Otherwise there will be a lot of unknown words, as the system will not find suitable tags for unknown words.

1.3.2 POS tagger

A Part-Of-Speech Tagger (POS Tagger) is a kind of software, which reads a text from a particular language. Then the tagger assigns POS tags to each word and other token of that text, such as noun, verb, adjective, adverb punctuation etc.POS taggers not only give POS information, they provides inflectional and lexico-semantic information. In this process, the tagger takes a word or sentence as input and as output; it tags the word or the words in that sentence with their morphological category, known as part of speech (POS). Taggers use various kinds of information source e.g. rules, lexicon and dictionaries. In dictionaries, category or categories of words are given. There are words which have more than one category. For example, in Sanskrit language *Bhavati* is both pronoun and verb. Rules or probabilistic information are used by the taggers to solve this ambiguity.

Dictionaries, lexicons, rules and other methods are used by taggers for solving the ambiguity. A category or several categories of words are given in dictionary. It is possible that a word may belong to more than one category. For example, word ‘run’ belongs to two category; noun and verb. Probabilistic information is also used to solve this ambiguity.

It is needed that a tagger should be robust, efficient, accurate, tuneable and reusable.

1.3.3 Why POS tagger?

Part of speech tagger, with good accuracy is the first need of many NLP tools. If a tagger with good accuracy has been used, then accuracy of other advance tools like Grammar checker, Phrase chunker, Machine Translation tool, become very efficient.

- **Speech synthesis and recognition-** As POS gives significant amount of information about the word and its neighbours, this become useful for development of language model for speech recognition. How a word is pronounced can be comprehended by the POS of that word depending upon the grammatical category of that word. For example the word object is pronounced as Object, if the grammatical category is noun. On the other it pronounced as object when it is tagged as verb

- **Information retrieval and extraction-**

  When a query is given to a information retrieval system with POS information then the system extract a refined information. For example, if a user gives a query for the documents containing "book" as noun, then if the user add the POS with the query then the information retrieval system generate all the documents with book as noun and avoiding the documents which has "book" as verb. Some Patterns used for information extraction from text use POS reference.

- **Machine translation-**

When translation from one target language to other source language is aimed then that case the POS category of the target word and source word becomes really important. Eg. the word 'kṛti' of Sanskrit can be translated both as noun and adjective. By giving POS category it can be easily decided.

A POS Tagger plays a huge role in the better performance of the systems related with speech reorganisation; parsing and information retrieval. Because POS tagger removes the ambiguity of sentences, by this it make these system easy to parse. In tools made for spelling correction; query answering, machine translation, searching large text databases, and information extraction, POS plays a significant role.

Brill tagger, Tree tagger, CLAWS tagger, online tagger ENGTWOL is some of the classic example of POS tagger. Indian languages also have many POS taggers which have been developed from last three decades.

1.4. Problems during Part of speech Tagging

The problem which arise during part of speech tagging are following :-

- **Ambiguity**- The main problem in part-of speech tagging is ambiguity. It is possible that a word in a sentence can act as more than one meaning so it can have more than one tag so such situation arise the problem of Ambiguity. To solve this problem we consider the context instead of taking single word.

- **Corpora**- For pos tagging large quantity of data are required. It become difficult with less-resourceful languages like Manipuri, Awadhi etc. as these language are not much computerized.

- **Linguistics expertise**- some time while training deciding a pos tag for certain word becomes an issue. Annotator- agreement is the solution for this problem.

- There can be variation in the size of the tagset and the ambiguity rate, depending upon the language to language. This variation can be problem specific too.

- There are some languages, which have a richer morphology than other languages. In this situation, the tagger needs an elaborated set of features, to handle those languages.

1.5. Features useful for POS Tagging[10]

The features which have been found useful during POS tagging are,

- **Suffixes**: A suffix is an affix which comes after the stem of a word. Common examples are case endings, which indicate the grammatical case of nouns or adjectives, and verb endings, which form the conjugation of verbs. This is a important feature which is is useful in tagging

- **Prefixes:** A affix which comes before the stem of the word, is called prefix. Adding a prefix to the beginning of one word changes it into another word. In the study of languages, a prefix is called a preformative, because it alters the form of the words to which it is affixed. The knowledge of prefixes makes pos tagging work easier.

- **Context Pattern based Features:** Context patterns are helpful for POS tagging. Eg.. Word prefix and suffix context patterns.

- **Word length**: Length of particular word is useful feature.

[10] Neetu Aggarwal, Amandeep kaur Randhawa. (2015). '*A Survey on Parts of Speech Tagging for Indian Languages',* International Journal of Computer Applications, International Conference on Advancements in Engineering and Technology.

- **Static Word Feature**: The previous and next words of a particular word are used as features.
- **Presence of Special characters**: Presence Special characters surrounding the current word are used as features. Like in when a question mark is available in a sentence, it becomes easy to recognise Wh- word.

1.6. Evaluation Methods[11]

The evaluation metrics for the data set is **precision, recall and F-Measure**. These are defined as following:-

- **Recall** = Number of correct answer given by system / Total number of words.
- **Precision** = Number of Correct answer / Total number of words.
- **F-Measure** = Recall *Precision / Recall + Precision

1.7. Tag-sets

Tagset is the set of tags which are used by tagger to assign to the relevant word. For developing a tagger a standard tagset is the basic requirement. Tagsets has two catagories coarse and fine grained. In the coarse tagset tags are simple e.g. N (Noun), V(Verb), ADJ (Adjective), ADV (Adverb), PREP (Preposition), CONJ (Conjunction)

On the other hand fine-grained tagsets have tags such as N_NN (Noun-Nominative), N_NNP (Noun-Proper Noun), V_VM_VF(Verb-Main Verb- Finite Verb), and so on. The taggers generally use fine-grained tagset, as they cover deep linguistic information.

1.7.1. Tagset development and design

Morpho-syntactic tagset which is popularly called as POS tagset establishes a link between morphological and syntactic levels of description. It also contains the orthographic extra-linguistic tags along with other symbol tags seen in the written text. Tags in a tagset differ from language to language owing to the nature, method of descriptive linguistic tradition applied and the orthographic representation (script) used in the specific language. It also

[11] Garg Navneet, Vishal Goyal, Suman Preet, (2012). *'Rule Based Hindi Part of Speech Tagger'*, Proceedings of COLING. pages 163–174, COLING 2012, Mumbai.

depends upon the purpose and the kind of information application needs in which the tagset is supposed to be used.

1.7.2. Design principles

Annotation is actually an interpretative process as it depends upon the set of symbols used to annotate the text which represent an analytical scheme which may have some theory behind it. It is generally considered to have a theoretically neutral approach to develop the set of symbols used in annotating the corpus. The set of symbols so developed should be consensual classes, uncontroversial so that it is used by many analysts.[12]

1.7.3. Features or Attributes

Feature or attribute tags contain the information like gender (male, female, neuter), number (singular, dual, and plural), case (nominative, accusative etc.), person as in the case of verbs (first, second, third) etc.

1.7.4 Classification based on tag structure

In the **flat Tagset** there is no clear structural distinction between major word class information and extra feature information. They have large number of tags, with their separate labels. It is not possible to scale any structure from these tags. But they are easy to build. The famous C5 Tagset used for annotating the BNC Corpus is flat tagset.

Hierarchical Tagsets contain less number of tags compared to flat tagset. They are modular, related structurally, easily scalable, requires more research to develop, and can be compared to related object oriented research.

1.7.5 Techniques of tagging

There are various tagging techniques used by different languages according to the, grammar data and other linguistic or NLP needs. Few of them are mentioned here :-

[12] Leech, Geoffrey and Smith, Nicholas, 1999*, 'The use of Tagging' in 'Syntactic Word-class Tagging'*, Kluwer Academic Publishers, Dordrecht, The Netherlands, p.24

1.7.5 .1. The linguistic Technique

In this technique hand-coded linguistic rules are applied. For this an expert linguist is required, who can formalize the restrictions of the language. This technique needs a huge cost and lot of time. It is also very much dependent on each specific language. The benefit of the linguistics approach is that for constructing language model a linguistic point of view is taken care of. The other advantage is that this technique has many and complex kind of knowledge of language.

1.7.5.2. The Learning Technique

When tagger used learning technique it derived rules from corpora (labelled or non-labelled). n-grams is the most extended formalism approach. In this technique the language model can be developed by using a tagged corpus, which is called supervised method. It also can be constructed with un-annotated corpus. In supervised methods; the model is trained by the relative observed frequencies. an initial model which is estimated using labelled corpora is used in unsupervised method in which the system learns using the Baum-Welch algorithm. The benefit of using learning technique is that building language model is easy. There is a flexibility of choice of categories. Adopting system for other language is easy in this method.

1.7.6 Corpora and Textual Annotation

Corpus is a collection of more than one text of a language from varied subjects and authors, which is a representative of that language, of finite size, which is machine readable with a standard reference. Annotated Corpora is the body representation of a language enhanced with various types of linguistic information[13]. According to Crystal (1992), "Linguistic corpus is a collection of data, either in written form or in the form of recorded speech". A corpus gives understanding of particular language to grammarians, lexicographers etc. Computer-processable corpora give opportunity to linguists to adopt the principle of total accountability. They can retrieve all the occurrences of a particular word or structure for inspection or randomly selected samples. One gets accesses to that lexical information, morphosyntactic information, semantic information and pragmatic information of a language by analysing corpus of that language.

[13] Corpus Linguistics, Lancaster, 'http://bowlandiles.lancs.ac.uk/monkey/ihe/linguistics/corpus2/2fra1.htm' (accessed: 15.07.15)

A corpus can be categorised in two categories; unannotated, annotated. An unannotated corpus is raw and simple text.The linguistic information is implicit in the unannoted corpus.In annotated corpus linguistic knowledge is explicitly induced.

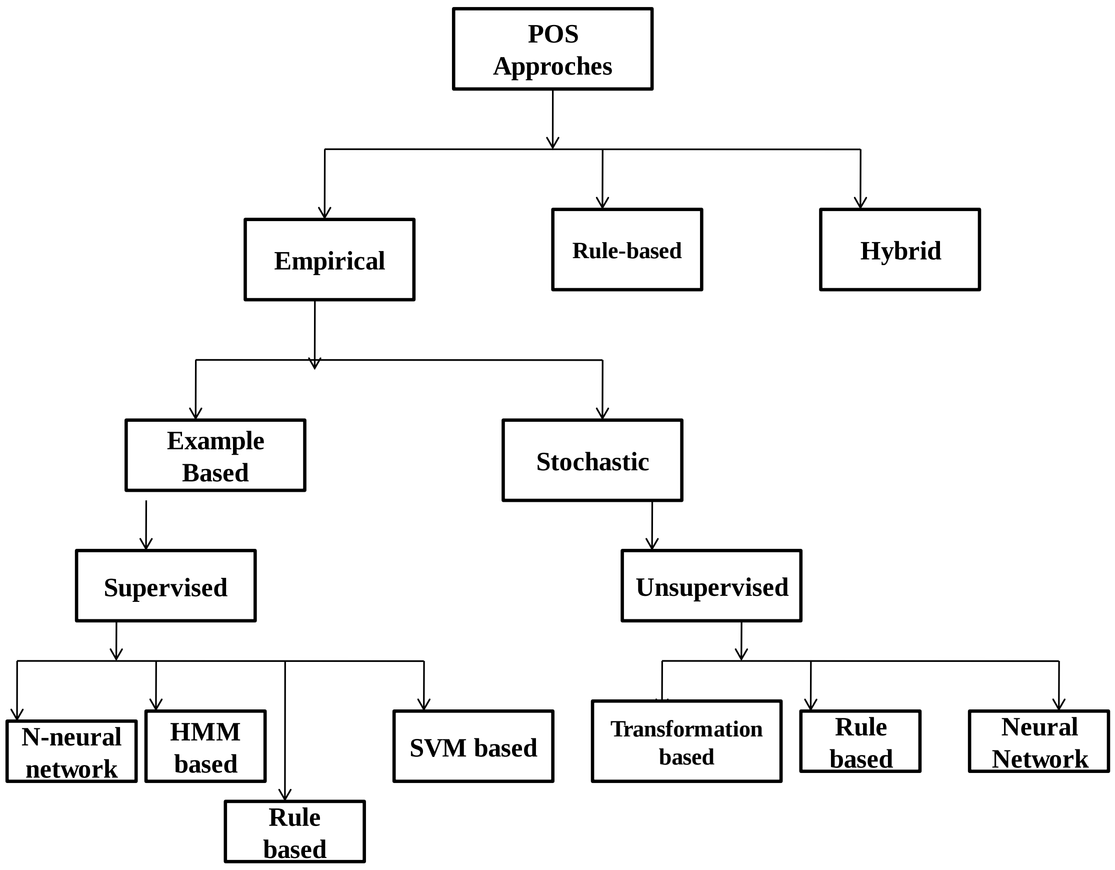
any descriptive or analytic notations which is applied to raw language is covered by `Linguistic annotation'. The basic data may be in the form of text. Transcriptions of all sorts from phonetic features to discourse structures are included in the added notation. These notions also include, POS and sense tagging, syntactic analysis, "named entity" identification, co-reference annotation, and so on[14].

For example, the form *yatante* contains the implicit POS information - third person plural present tense of the verb root '*yat*'- but it is only comprehendible in normal reading by recourse to the user's knowledge of Vyākaraṇa. However, in an annotated corpus the form *yatante* might appear as "यतन्ते [A_laT_1.3]", with the code A_laT_1.3 indicating that it is a *prathama puruṣa bahuvacana* (1.3) present tense (laT) form of a lexical verb in the *ātmane* termination (A). this kind of annotation makes it fast and easy to retrieve information and analyze the language contained in corpus.[15]

[14] Linguistic Annotation, UPENN, 'http://www.ldc.upenn.edu/annotation'/ (accessed: 15.07.15)
[15] R. Chandrashekar, (2007), ***Part-of-Speech Tagging for Sanskrit,*** Ph.D. Thesis, JNU, New Delhi

1.8 Classification of POS tagging approaches

this figure gives an overview of different pos tagging approaches.



Part of speech tagging methods can be categorized in three parts.

1.8.1. Rule based Tagging

The rule based POS tagging approach is based on a set of hand written rules. These rules are created with the help of grammarian. These rules considered on the basis of the morpheme ordering and contextual information. It is oldest tagging approach used by initially developed taggers. Rule based technique use contextual information for assigning a POS tags to a particuler word. These rules are called as context frame rules. For example of a context frame rule might say something like: "If an ambiguous/unknown word X is preceded by a Determiner and followed by a Noun, tag it as an Adjective".[16] These rules need to be written with a help of an expert and checked properly.

[16] Antony P J, Dr. Soman K P,(2011).' *Parts Of Speech Tagging for Indian Languages*: *A Literature Survey'*, International Journal of Computer Applications (0975 – 8887) Volume 34– No.8

"Brill's tagger" is the first and extensively used English POS-taggers which is based on rule based algorithms. Two-stage architecture was used in initial algorithms for assigning POS to word. The first stage, there is a dictionary which is used to assign each word a list of potential parts of speech. The second stage contains large lists of hand-written disambiguation rules. These rules were there to narrow down those lists to assign a particular POS for each word. The ENGTWOL tagger has followed similar two stage architecture.

The advantage of rule based machine POS approach is that it does a deep analysis of syntax and semantic levels. Requirement of huge linguistic knowledge is the major drawbacks of this approach. It also need very large number of rules to cover all the features of a language. This approach is labour-intensive and costly.

1.8.2. Empirical Tagging[17]

Empirical tagging technique is also called corpus based tagging. In this technique rules are generated from a given corpus. According to P.J Antony e.l. (2011); majority of the taggers belong from empirical category. The emergence of this method is caused by theses reasons: -

- The increasing availability of machine readable text
- The relative failure of rule-based approaches
- The increase in capability of hardware (CPU, memory, disk space) with decrease in cost are some of the reasons

The empirical approach of parts speech tagging is further divided in to two categories:

- Example based approach
- Stochastic based approach

Literature shows that majority of the developed POS taggers belongs to empirical based approach

[17] Ibid

1.8.3. Stochastic based POS tagging

Stochastic approach required a sufficient large sized corpus. It finds out the most frequently used tag for a specific word in the annotated training data and uses this information to tag that word in the un-annotated text. In this method, frequency, probability or statistics of the word occurred in the training corpus get calculated by the tagger. The problem which comes in this approach is that, the tagger can generate the sequence of tags for words that are not acceptable by the linguistic expert of that language as they do not follow the grammar.

Bahl and Mercer (1976) made the first tagger based on probabilities in tagging, with Viterbi decoding. Some of the other stochastic taggers are Marshall, 1983; Garside, 1987; Church, 1988; DeRose, 1988. Stochastic based approach further divided in two categories; Supervised and unsupervised.

1.8.4. Supervised Tagging-

In this method supervised tagging, a corpus is used which is already annotated. This pre-trained corpus is get utilized for training of the tagger. The tagger learns information from this tagged training data. In this learning tagger learn about the tagset, the frequencies of word-tag and rule sets etc.

The benefit of the supervised tagging is that accuracy of the results is high. The main draw back with this method is that tagging of training data is a costly and time-consuming process.

1.8.5. Unsupervised Tagging-

The unsupervised POS Tagging methods do not require a corpus which is pre-tagged. On the other hand they use some advanced computational techniques. One of these techniques is the Baum-Welch algorithm. By using these techniques the tagger automatically generate tagsets, transformation rules, etc. By using this information, they induce the contextual rules or calculate the probabilistic information, which is needed by rule based system or transformation based system.

The problem with this method is word clustering. Because of that results from this method are very coarse. In this process many fine distinctions remains left out, which can be easily found in the supervised methods.

1.8.6 Transformation-based POS tagging[18]

c is a rule-based algorithm. It is used for automatic tagging of parts-of-speech to the given text. TBL automatically induce the transformation rules from corpus. This approach is a combination rule based and stochastic. TBL is similar to the rule based taggers, as it is based on rules. These rules specify what tags should be assigned to a particular word. TBL is a machine learning technique, because it automatically learns rules from data like stochastic taggers.

Using transformation rules, TBL transforms one state to another, in order to find the most suitable tag for each word. TBL provides user linguistic knowledge in a readable form. It automatically learns linguistic information of that language from corpora. In this approach automatic rule induction is to run an untagged text through a tagging model and get the initial output. Then a annotator goes through the output of this first phase and corrects those tagged words by hand, which were not correctly tagged. After that the validated data is given to the tagger. The tagger then learns correction rules by doing a comparison of two sets of data. This process need repetition several times before the tagging model can achieve considerable performance. Transformation-Based Tagging is also known as Brill tagging. This approch is an example of the Transformation-Based Learning (TBL), which is a machine learning approch (Brill, 1995). TBL is a mixed bag of both the rule based and stochastic technique.

A typical transformation-based learner has three components; an initial state annotator, a set of transformations rules and an objective function.

1.9 Machine learning (ML)

Machine learning is a subfield of computer science. ML has evolved from the study of computational learning theory in artificial intelligence and pattern recognition. Construction and study of algorithms that can learn and predict from data, are the basis of ML. It is inherently a multidisciplinary field i.e. artificial intelligence, computational complexity theory, probability, statistics, information theory, psychology, and neurobiology, philosophy. It is a diverse and exciting field, and there are multiple ways of defining it. [19]

[18] Ibid

[19] http://www.aaai.org/AITopics/html/machine.html (accessed: 23.07.15)

A system which is capable of the autonomous acquisition and can integrate knowledge is referred as ML system. This system has capability to do analytical observation, learn from experience and other ways. This capacity make it able to improve and increased its efficiency.

ML is the study of computational systems that improve performance on some task with experience as Mitchell (2006) defines,

"--- a machine learns with respect to a particular task T, performance metric P, and type of experience E, if the system reliably improves P at task T, following experience E."

In a further explanation Schapire (2005) explains it further, that ML studies how to automatically learn to make exact predictions based on past observations. The primary goal of ML is to according to Schapire (2005) is to classify new examples of the test data into given set of categories with highest accuracy

In simple word machine learning is the science of getting computers to act without being explicitly programmed and concerned with the question of how to construct computer programs that automatically improve with experience.

1.9.1 Objective of Machine Learning

The objective of the machine learning is to make machine capable of learning the intelligent abilities of humans. It searches for patterns and relationship in data that can be modelled. Large set of annotated data is used to train a model, which predicts it on test data. Bogers (2005) describes four stages of this process:

(i) Feature extraction- observing the different features in the data that can be used for learning. For example, for NE learning the features like surrounding word, POS category of word etc.

(ii) Feature selection- deciding that which features have the optimal predictive value for a given problem;

(iii) Algorithm selection- algorithm of ML is selected.

(iv) Labelling decisions- how to label the tags in training data

1.9.2 Method of ML

Machine learning methods can be categorised in two phases:

- **Training:** A ML system is learns from a set of **training data**.
- **Application:** The model is applied to make decisions about some new **test data**.

For example, in the spam filtering case, the training data constitutes email messages labelled as ham or spam, and each new email message that we receive (and which to classify) is test data

1.9.3 Advantages and Disadvantages of Machine Learning

Describing the advantages of ML approach, Schapire (2005) says that it is easier than RB approach because it is mainly corpus-based approach. It gives automatic method to search for hypotheses explaining data. It is flexible because it can be applied to any learning task. Another advantage is that humans often are incapable of expressing the rules of the language but they can easily classify examples. Armstrong-Warwick (1993) (as cited by Brill and Mooney, 1997, p.16) describes that empirical methods offer potential solutions to solve several problems in NLP through:

- Acquisition, it allows learning of relevant knowledge from data rather than laboriously hand coded rules
- Coverage, an extensive and comprehensive training data set will give good result on test data;
- Robustness, it allows selection of the best option and thus increases the robustness of the system
- Extensibility, by allowing using the system on additional data or data of new domain. In addition, Church and MERCER (1993) (as cited by Brill and Mooney,

1997, p.16) describe three recent developments which have also helped in the advancement of this field.

However, it has disadvantages also as it needs a lot of labelled data; it is error-prone and getting perfect accuracy is usually impossible. Another disadvantage is that it is difficult to know what was learned. This is why this approach is known as ‘knowledge poor approach.”

1.9.4 Rationale behind use of Machine Learning in POS Tagging

According to Church and MERCER (1993) (as cited by Brill and Mooney, 1997, p.16) describe three recent developments which helped the expansion of machine learning uses in NLP and specially POS tagging. They are :-

- Computing resources- the availability of sufficient processing and memory resources to analyze large amounts of data;
- Data resources- the development and availability of large corpora for training and testing systems; and
- Emphasis on applications and evaluation- the focus in industry and government is on the development of practical systems that can be evaluated on real data.

1.9.5 Machine learning approaches used in POS tagging[20]

With the advancement of technology machine learning has evolved gradually. Many techniques of ML are used by different field of AI. Use of these technique in natural language is very frequent. There are several prominent approaches used in the POS tagging.

1.9.5.1. Uni-gram-

The unigram (n-gram, n = 1) tagger is the basic statistical tagging algorithm. In this algorithm, the tagger assigns most suitable tags to each token. For example, it will assign the tag JJ to any occurrence of the word *Madhura*, since *Madhura* is used as an adjective (e.g. a *Madhuram Falam*) more often than it is used as a Proper noun (e.g. *Tasya nāma madhura iti asti*). A training corpus is needed for the training of the unigram tagger to learn and then tag

[20] Hasan, Fahim Muhammad, Naushad UzZaman and Mumit Khan,(2007). ‘*Comparison of different POS Tagging Techniques (n-Gram, HMM and Brill’s tagger) for Bangla’,* Center for Research on Bangla Language Processing, BRAC University, Bangladesh

the data. The corpus is used to determine the frequency of the tag for each word. After that a default tag will be assigned to that word. There will be no tag is given by the tag to a word, if there is not any token in the training data.[21]

1.9.5.2. Hidden Markov Model (HMM)

The intuition behind HMM (Hidden Markov Model) and all stochastic taggers is a simple generalization of the "pick the most likely tag for this word" approach. The unigram tagger only considers the probability of a word for a given tag t; the surrounding context of that word is not considered. On the other hand, for a given sentence or word sequence, HMM taggers choose the tag sequence that maximizes the following formula: P (word | tag) * P (tag | previous n tags).

1.9.5.3. Support vector machine (SVM)[22]

SVM is a machine learning algorithm for binary classification, which has been successfully applied to a number of practical problems, including NLP.

Let {(x1 , y1). . . (xN, yN)} be the set of N training examples, where each instance xi is a vector in R N and yi ∈ {−1,+1} is the class label. In their basic form, a SVM learns a linear hyperplane, that separates the set of positive examples from the set of negative examples with maximal margin (the margin is defined as the distance of the hyperplane to the nearest of the positive and negative examples). This learning bias has proved to have good in terms of generalization bounds for the induced classifiers. The SVMTool is intended to comply with all the requirements of modern NLP technology, by combining simplicity, flexibility, robustness, portability and efficiency with state–of–the–art accuracy. This is achieved by working in the Support Vector.

1.9.5.4. Neural Networks

The neural network approach has been one of the most explored approach in the Artificial Intelligence (AI) field. There are various rules used for the process of learning in neural network. The parameter change is the key to determine the type of learning in neural

[21] Ibid.
[22]Chachoo Manzoor Ahmad, S. M. K. Quadri, 2012. '*Adaptive Hybrid POS Cache based Semantic Language Model'*, International Journal of Computer Applications (0975 – 8887) Volume 39– No.13.

network. This can be done with or without an instructor. The neural networks can be categorized in three groups:

1. supervised learning
2. unsupervised learning
3. reinforcement learning

There is a similar model, called semantic network. In semantic networks, there are nodes that are the representation of the concepts and connections. These nodes represent the semantically meaningful association between these concepts. Associative network models is the better characterization rather than neural/brain models. The activation rules that implement information retrieval in these associative networks, often referred to as spreading activation, typically produces an intersection search. Hence, they are also called “spreading activation” models (Doszkocs, Reggia and Lin, 1990).[23]

1.9.5.5. Memory Based Learning (MBT)

MBT is a memory-based tagger-generator and tagger in one. It is based on Memory Based Learning. The tagger tags new sequences and the tagger-generator generates a sequence tagger on the basis of a training set of tagged data. MBT can be used as part-of-speech taggers or chunkers for NLP. It can also be used for domain specific named-entity recognition (NER) and information extraction and disfluency chunking in Speech to text.[24]

Other models

1. Stanford POS Tagger (Toutanova et al. 2003)

2. Multi-class SVM model which is trained by SVM- Multiclass (Joachims 1999)

3. Stochastic gradient descent (SGD) by Shalev-Shwartz et al. (2007).

[23] Hasan, Fahim Muhammad, Naushad UzZaman and Mumit Khan,(2007). ‘*Comparison of different POS Tagging Techniques (n-Gram, HMM and Brill’s tagger) for Bangla’,* Center for Research on Bangla Language Processing, BRAC University, Bangladesh

[24] http://ilk.uvt.nl/mbt/ (accessed: 31.07.15)

1.9.5.6. Transformation-Based Learning

Transformation-Based Learning (TBL) which is a symbolic Machine Learning method is developed by Brill (1995). According to Atwell et.al.(2000), ”In TBL method a tagged training corpus is given to the system, then the system produces a sequence of rules that serves as a model of the training data. To derive the appropriate tags, each rule may be applied in order to each instance in an untagged corpus.TBL relies heavily on a large annotated training corpus, and relies on reasonable default heuristics to get things started. It learns rules that are easily understandable and allows rules to be easily acquired for different domains or genres. As mentioned above, TBL has been widely used for Part-of-Speech tagging”[25].

There is a gap between an initial semantic network generated from input data, and a semantic one representing profound knowledge, from which a knowledge database can be constructed. By using transformation rules, the semantic analysis method is based on a pattern matching with a semantic network. A transformation rule description language allows users to manipulate their knowledge base and to define rules.

1.9.5.7. Decision Tree classification

A decision tree is constructed by recursively partitioning the training set, selecting, at each step, the feature that most reduce the uncertainty about the class in each partition, and using it as a split. For example, (Cohen 1995) used the decision tree learner Ripper to induce a decision tree that was used to automatically label a new corpus with predicates, and used 5-fold cross validation to ensure results were stable. [26]

1.9.5.8. Maximum Entropy Model (MEM)

The Maximum Entropy Model is based on the principle of Maximum Entropy. This states that while selecting among a number of different probabilistic models for a set of data, the one which makes fewest arbitrary assumptions about the nature of the data is the most valid

[25] Atwell E, Demetriou G, Hughes J, Schiffrin A, Souter C, and Wilcock S, (2000). ’*A comparative evaluation of modern English corpus grammatical annotation schemes’*. ICAME Journal, volume 24,pages 7-23, International Computer Archive of Modern and medieval English, Bergen

[26] Rose Hu Xunlei, Eric Atwell, (2003). ‘*A survey of machine learning approaches to analysis of large corpora’***,** School of Computing, University of Leeds, U.K. LS2 9JT

model.[27] It was first introduced by Ratnaparkhi (1996) and McCallum et. al (2000). The model probability of history *H* with tags *T* is defined as:

$$p(h,t) = \pi\mu \prod_{j=1}^{k} \alpha_j^{f_j(h,t)}$$

Equation 1

In the above equation {a1, ..., ak} and {f 1 , . . , fk} are the positive model parameters and ‘features’, respectively.

where,

fj(h, t) = {0, 1} and parameter *aj* corresponds to a feature *fj*.

The chosen parameter to maximize the training data p is {a1, …, ak} (see Karthik, K.).

1.9.5.9. Conditional Random Fields (CRF)

A Conditional Random Field (CRF) segments and labels a sequence of data on the basis of a probabilistic framework. A conditional model specifies the probabilities of possible label sequences given an observation sequence. This probability of the label sequence may depend on arbitrary or non-independent features of the observation sequence. The probability of a transition between labels may depend not only on the current observation, but also on past and future observations. The CRF model calculates the probability based on some features, which might include the suffix of the current word, the tags of previous and next words, the actual previous and next words etc. [28]

[27] MacKinlay Andrew (2005). *“The Effects of Part-of-Speech Tagsets on Tagger Performance”. Ph.D.Thesis* Submitted to The Department of Computer Science and Software Engineering ,University of Melbourne.

[28] Hasan, Fahim Muhammad, Naushad UzZaman and Mumit Khan,(2007). ‘*Comparison of different POS Tagging Techniques (n-Gram, HMM and Brill’s tagger) for Bangla’*, Center for Research on Bangla Language Processing, BRAC University, Bangladesh

CHAPTER 2:

SURVEY OF POS TAGGERS

Introduction:-

This chapter gives a detailed survey of the taggers developed till date. The first part of the chapter gives a brief history of part of speech tagging. In this part taggers developed around the globe, has been discussed. The second part of the chapter gives detailed overview of the taggers developed in Indian languages. Taggers in each language have given in a wide illustration. Third part of the chapter contains tagger description of less resourceful languages. Last part deals with the description of POS taggers in Sanskrit.

2.1. A Brief History of Part-of-Speech Tagging

Existing taggers can be classified into three main groups according to the kind of knowledge they use: linguistic, statistical and machine-learning family.

Within the linguistic approach most systems codify the knowledge involved as a set of rules (or constraints) written by linguists. The linguistic models range from a few hundred to several thousand rules, and they usually require years of labour

Automatic taggers developments started in late 1950 and 1960. These taggers used hand-coded disambiguation rules, in the form of regular expressions compiled into finite-state automata. The lexicons were small in these programs. It gave all the possible analyses to the input words. Heuristic rules like affix letter sequence analyses, capitalization, and grapheme clues about the category were used to tackle lexical absence problem.[29]

The first system which has gone to the record is a Finite-State Parser created at University of Pennsylvania (UPENN) in 1958-59. This was the first application of Finite State Transducers (FST) to parsing. The lexical ambiguities were resolved using rule based disambiguation techniques using a grammar of 14 ordered context rules.[30]

In 1963, Klein and Simmons developed Computational Grammar Coder (CGC) which had three components viz. a lexicon, morphological analyzer (MorphA) and context

[29] Voutilainen, Atro, 1999, 'A Short History of Tagging' in Syntactic Wordclass Tagging, Ed. Halteren, Hans van, Kluwer Academic Publishers, Netherlands p.10
[30] Joshi, A. and Phil Hopely, 1997, A parser from antiquity. http://www.cis.upenn.edu/~cse477/fst-parser-uniparse.pdf (accessed: 25.10.2015)

disambiguator. This system has reported 90% accuracy on applying a 30-tag to articles from the Scientific American and a children's encyclopaedia.[31]

On the basis of CGC system, Greene and Rubin in 1971 developed a tagger named TAGGIT. This has a lexicon and a tagset of 87 tags. The application was done on the Brown Corpus (American English Corpus).It has tagged 77% of the corpus with accuracy. The rest of the corpus was tagged manually.[32]

The statistical approach came like a wave in POS tagging arena as it needed less human effort. In this approach focus is on building a statistical model of the language and using this model to disambiguate a word sequence. The language model is coded as a set of co-occurrence frequencies for different kinds of linguistic phenomena. It was a breakthrough.

In late 1970, the Lancaster-Oslo/Bergen (LOB) corpus was build. This was a British English Corpus based on the same method on which Brown corpus was created. This used a tagger CLAWS1[33]. The algorithm used by this tagger was probabilistic. This tagger was the probabilistic version of TAGGIT. This algorithm used tag bigram probabilities, but instead of storing the word-likelihood of each tag, tags were marked either as 'rare', 'infrequent', or 'normally frequent'.[34]

The second significant tagger with probabilistic algorithm was developed by Church K.W. in 1988 which was very close to HMM tagger. It was an extension of CLAWS tagger with Viterbi decoding algorithm which was used to find a tag sequence.

1990s was the era of explicit use of hidden Markov model. Kupiec, J (1992), Merialdo, B (1994), Weischedel et al. (1993) were some taggers which used HMM with EM training algorithm.

In the recent time all the stochastic algorithms use various statistical and machine-learning tools for predicting the probability of a tag or tag-sequence. These algorithms use a large

[31] Jurafsky, Daniel & James H. Martin, 2002, 'Speech and Language Processing', Pearson Education, Delhi, p.318

[32] Ibid.

[33] CLAWS1 (Constituent-Likelihood Automatic Word Tagging System, version 1) was the first tagger to use a probabilistic algorithm, which disambiguates by choosing the correct word category on the basis of statistical corpus evidence. This approach is viewed as an approximation to HMM tagging approach. To be precise, it is a kind of 'open' Markov model approach. Presently CLAWS has reached version 4, which gives 96-97% accuracy. UCREL offers free CLAWS WWW trial service at http://www.comp.lancs.ac.uk/ucrel/claws/trial.html (accessed 16.10.2015)

[34] Ibid. p.318

number of information like what neighbouring words are, what their POS are, and also the orthographic and morphological features. These features are then combined to estimate the probability of tag – through *decision tree* as used by Jelinek et al. (1994), Magerman (1995); - through Maximum Entropy algorithm as used by Ratnaparkhi (1996); - through log linear models as used by Franz (1996); - through networks of linear separators (SNOW) as used by Roth and Zelenko (1998). Brill uses a rule based approach with the unsupervised version of TBL algorithm.[35]

Here is a brief review of some notable POS taggers-

2.1.1 TnT -- Statistical Part-of-Speech Tagger

Trigrams N Tags (TNT) is a stochastic HMM tagger. It is based on trigram analysis, which is developed in Saarland University in year 1993-1999 by Thorsten Brants. The notable feature of TnT is to uses a suffix analysis technique. This technique is based on properties of words like suffixes in the training corpora. By this approach the system estimate lexical probabilities for unknown words that have the same suffixes. TnT is language independent as it is not optimized for any specific language, so this tagger is trainable on different languages and virtually any tag set. The tagger is an implementation of the Viterbi algorithm for second orders Markov models.

2.1.2 IMS's TreeTagger

This tagger was developed as a part of the project *Textcorpora und Erschließungswerkzeuge* (1993-1996) at the IMS (same project as Corpus Workbench, CWB/CQP).This is a language-independent POS tagger. It is available free for academic use and comes with free language models for approximately 10 languages. It is not open source and its assessment is poor. This is a HMM tagger using decision trees for smoothing. Documentation is sparse (included as a *readme* in the download package).

2.1.3 Stanford Log-linear Part-Of-Speech Tagger[36]

It is open source and models for English, Arabic, Chinese, and German. The java implication is used in this tagger. The tagger was originally written by Kristina Toutanova. This tagger is based on the Maximum Entropy framework. It can be trained on any language on a POS-

[35] Ibid., pp.318-319
[36] http://nlp.stanford.edu/software/tagger.shtml (accessed 26.10.2015)

annotated training text for the language. The best resulting accuracy for the tagger on the Penn Treebank is 96.86% overall, and 86.91% on previously unseen words.

2.1.4 Brill's rule-based POS tagger[37]

Eric Brill introduced a POS tagger in 1992 that was rule- based. In this tagger the grammar is induced directly from the training corpus without human intervention or expert knowledge. The only additional component necessary is a small, manually and correctly annotated corpus - the training corpus - which serves as input to the tagger. The system is then able to derive lexical/morphological and contextual information from the training corpus and 'learns' how to deduce the most likely part of speech tag for a word. Once the training is completed, the tagger can be used to annotate new, un-annotated corpora based on the tagset of the training corpus.

2.1.5 Chris Biemann's unsupos – unsupervised POS tagging[38]

It is an open source tagger. It is a model for a number of languages. In this tagger unsupervised POS tagging has been used. It does not require an annotated training corpus. Instead, word categories are determined by analyzing by a plain text which is monolinguial and sentence-seprated. The architecture of this system is easy to integrate in various environments.

2.1.6 MBT: Memory-based tagger generation and tagging

MBT is a memory-based tagger as well as generator in one system. The tagger-generator of the system can generate a sequence tagger, which is based on the training set of tagged sequences.

The tagger part of this system can tag new sequences. Because of that **MBT** can, for instance, be used to generate part-of-speech taggers. It can also used as chunkers for natural language processing. It has also been used for named-entity recognition, information extraction in domain-specific texts, and dissiliency chunking in transcribed speech.[39]

[37] http://en.wikipedia.org/wiki/Brill_tagger (accessed 26.10.2015)
[38] http://wortschatz.uni-leipzig.de/~cbiemann/software/unsupos.html(accessed 26.10.2015)
[39] http://ilk.uvt.nl/mbt/(accessed 26.10.2015)

2.1.7. CLAWS (Constituent Likelihood Automatic Word Tagging System)[40]

It is an Open source tagger which has Unsupervised POS tagging. It does not require an annotated training corpus. Instead, word categories are determined by analyzing a large sample of monolingual, sentence-separated plain text. The tag set can probably not be determined by the user/linguist. This tagger has no lemmatization. Tagger's code is written in Java. Its architecture is easy to integrate in various environments. In terms of usability documentation is sparse, homepage and maintenance do not seem to be quite up-to-date. [41]

2.1.8. Tree Tagger

A decision tree based tagger from the University of Stuttgart (Helmut Schmid). It is language independent, but comes complete with parameter files for English, Germen, Italian, Dutch, French, Old French, Spanish, Bulgarian,, and Russian. It is compatible to Linux, Sparc-Solaris Windows, and Mac OS X versions.

2.1.9. Apache UIMA Tagger

This tagger is open source. It is a good model for English and German. This HMM based tagger is a part of *Apache Unstructured Information Management Architecture* (UIMA) framework. The coding is done in JAVA, which is very good. The architecture of this tagger is flexible and Web service integration as component of the framework. [42]

2.2. POS taggers in Indian languages

India is a large multi-lingual, country with multi-ethnic culture. There are four main language families found in India, viz., Austro-Asiatic, Dravidian, Indo-Aryan and Tibeto-Burman, of which Dravidian and Indo-Aryan (IA) form the largest group of languages spoken in the sub-continent. Where there is huge work has already done in English and other languages, the NLP work in Indian languages has been started from last 3 decades.

[40] http://www.comp.lancs.ac.uk/ucrel/claws/trial.html (accessed 26.10.2015)
[41] http://wortschatz.uni-leipzig.de/~cbiemann/software/unsupos.html accessed on 16.07.2015
[42] http://uima.apache.org/sandbox.html (accessed 26.10.2015)

In 1990s Bharti et. Al was the first Indian who started working in the field of POS tagging.[43] They proposed a POS tagger for Hindi with morphological analyzer. The morphological analyzer in that tagger was used to provide a root word with its morphological features first and then a general POS category which can further classified using this generic pos category and morphological features. This tagger reported an accuracy of 78.66%.

For Bengali, Dandapat studied the possibility of developing a tagger using HMM and Maximum Entropy (ME) models. They too used a morphological analyzer for compensating the shortage of annotated corpus. A supervised tagger and a semi-supervised tagger has been developed with these two modes. An accuracy of around 88% has been achived with these two approaches. Ekbal and Bandyopadhyay annotated news corpus and developed an SVM based tagger[44]. They accuracy reported by their tagger was 86.84% for. Conditional Random Fields (CRF) based tagger was also developed by Ekbal in 2007. The information of prefix and suffix of Bengali words with normal word/tags was used for training the tagger. The tagger reported an accuracy of 90.3%.[45]

2.2.1. POS Taggers for Hindi

Hindi is the official language of India. About 182 million people speak Hindi as their native language and many others speak Hindi as a second language-some estimates say that around 350 million people speak Hindi.[46] Hindi is a morphologically rich language. Different POS tagging approaches have been proposed for Hindi Language. A tagging method for Hindi was proposed in that overcome the troubles in accurate tagging due to the scarcity of large sized training corpora.

The first initiative in POS tagging in Indian languages has been taken in hindi as above said.In last one decade there is a significant work has been done for developing taggers for Hindi by using different approaches. In 2006 Morphology driven, ME and CRF approaches

[43] Bharati, A., Chaitanya V. Sangal R., (1995) "Natural Language Processing – A Paninian Perspective". Prentice-Hall India, New Delhi .

[44] Ekbal, A., Bandyopadhyay, S., (2007*) "Lexicon Development and POS tagging using a Tagged Bengali News Corpus"*. In: FLAIRS-2007, Florida, pp 261-263.

[45] Ekbal, A., Haque, R., Bandyopadhyay, S., (2007) "Bengali Part of Speech Tagging using Conditional Random Field". In: 7th International Symposium of Natural Language Processing(SNLP-2007), Thailand Pattaya, 13-15 December 2007, pp.131-136.

[46] Dinesh Kumar and Gurpreet Singh Josan,(2010), "Part of Speech Taggers for Morphologically Rich Indian Languages: A Survey", International Journal of Computer Applications (0975 – 8887) Volume6–No.5, September, 2010, www.ijcaonline.org/ volume6/number5 /pxc3871409 .pdf..

based three taggers were developed. Then a HMM based tagger was developed by Manish Shrivastava and Pushpak Bhattacharyya in year (2008).

There are few POS taggers, developed in Hindi language by using different approaches. In 2006, Smriti Singh proposed a POS tagging methodology for resource-poor languages. The POS tagger was a rule-based tagger which is built based on hand-written morphology rules. The system used small and modestly-sized corpora of 15,562 words. But the system was exhaustive because of morphological analysis, which is backed by high-coverage lexicon and a decision tree based learning algorithm (CN2). For identifying the other POS categories, the system uses Lexicon lookup. The performance of the system was properly evaluated by a 4-fold cross validation over the corpora. This tagger has achieved significant accuracy of 93.45%. [47]

Aniket Dalal et.al. had used Maximum Entropy (ME) based approach for developing a tagger. ME approach requires feature functions extracted from a training corpus. This tagger showed that the performance of the system depend on size of the training corpus. The tagger performed well till it reaches 75% of the training corpus after that there was a reduction in accuracy. The reduction in the accuracy was due to over fitting of the trained model to training corpus. The highest accuracy was 89.34 %. Agarwal Himashu and Amni Anirudh were developed a Conditional Random Fields based POS tagger in 2006[48]. For training the tagger used a Hindi morph analyzer to get the root-word and possible POS tag for every word in the corpus. The training is performed with CRF++. Training data has information like suffixes, word length indicator and special characters. A corpus size of 1, 50,000 words were used for training and testing purposes and accuracy of the system was 82.67%. An improved Hindi POS tagger was developed by employing a naive (longest suffix matching) stemmer as a pre-processor to the HMM based tagger.[49] This method only required a list of possible suffixes which can be easily created using existing machine learning techniques for the language. No other linguistic resource was required for this method. The tagger reported performance of the system was 93.12%. Nidhi Mishra and Amit Mishra proposed a Part of Speech Tagging for Hindi Corpus in 2011. In the proposed method, the system scans the

[47] Ibid.

[48] Antony P J, Dr. Soman K P, (2011). *" Parts Of Speech Tagging for Indian Languages: A Literature Survey*", International Journal of Computer Applications (0975 – 8887) Volume 34– No.8,

[49] Manish Shrivastava and Pushpak Bhattacharyya (2008), *"Hindi POS Tagger Using Naive Stemming : Harnessing Morphological Information Without Extensive Linguistic Knowledge*", Department of Computer Science and Engineering, Indian Institute of Technology, Bombay. Proceeding of the ICON 2008.

Hindi corpus and then extracts the sentences and words from the given corpus. Also the system search the tag pattern from database and display the tag of each Hindi word like noun tag, adjective tag, number tag, verb tag etc. [50]

In year 2007, Based on lexical sequence constraints, a POS tagger algorithm for Hindi was proposed by Pradipta Ranjan Ray, Harish V., Sudeshna Sarkar and Anupam Basu. The proposed algorithm acts as the first level of part of speech tagger, using constraint propagation, based on ontological information, morphological analysis information and lexical rules. Even though the performance of the POS tagger has not been statistically tested due to lack of lexical resources, it covers a wide range of language phenomenon and accurately captures the four major local dependencies in Hindi [51]

2.2.2 POS Taggers for Bengali

Bengali, a member of the Indic group of Indo Iranian or Aryan branch of the Indo–European family of languages, originated from the eastern variety of the Magadhi Apabhramsa/Aavahatta. Bengali is a morphologically rich language. It is the seventh popular language in the world, second in India and the national language of Bangladesh. In the context of NLP, Bengali is has an edge. There is a substantial amount of work has already done in POS tagger developments for Bengali language using different approaches.

In the first attempt three different types of stochastic POS taggers were developed. In this attempt a supervised and semi supervised bigram HMM & a ME based model was explored based on tagset of 40 tags. A manually annotated corpus of about 40,000 words was used for both supervised HMM and ME model. For testing a set of randomly selected 5000 words have been used for all three cases and the results showed that, the supervised learning model outperforms over other models. A morphological analyzer can improve the performance of the tagger.

In 2011, Conditional Random Fields (CRF) framework where features selection plays an important role in the development of POS tagger was used for developing a tagger for Bangla

[50] Antony P J, Dr. Soman K P, (2011). *" Part of speech tagging for Indian languages : A Literature Survey",* International Journal of Computer Applications (0975 – 8887) Volume 34– No.8, November 2011
[51] Pradipta Ranjan Ray, Harish V., Sudeshna Sarkar and Anupam Basu, "Part of Speech Tagging and Local Word Grouping Techniques for Natural Language Parsing in Hindi",Department of Computer Science & Engineering, Indian Institute of Technology, Kharagpur, INDIA 721302. www.mla.iitkgp.ernet.in/papers/hindipostagging.pdf.

by Debasri Chakrabarti. A tagset of 26 tags were used to develop the POS tagger. For training purpose a corpus size of 72,341 tagged words were used. The system was tested with 20000 words selected from out of corpus and achieved 90.3%.

In the year 2007, two stochastic based taggers were proposed by Sandipan Dandapat, Sudeshna Sarkar and Anupam Basu using HMM and Maximum Entropy (ME) approaches.[52] Also Ekbal Asif developed a POS tagger for Bengali language using Conditional Random Fields (CRF).[53] In 2008, Ekbal Asif and Bandyopadhyay S developed another machine learning based POS tagger using SVM algorithm.[54] An Unsupervised Parts-of-Speech Tagger for the Bangla language was proposed by Hammad Ali in 2010.The tagger was based on a Baum-Welch trained HMM approach.[55]

2.2.3 POS Tagger for Odia

Odia is a morphologically rich language which possesses the salient features like PN and TAM being embedded in the verbs, serial verb constructions, ECV, causative constructions and conjunct verbs. Generally SOV word order is the most preferred one in sentence constructions while the possible word orders can be SVO and OVS constructions" (Jha, et al., 2014).

A Single Neural Network-based POS tagger for Odia language has been developed by Das and Pattnaik (2014). Initially, the tagger has been selected empirically 'with a definite length of contextual information'. After that, multiple neurons comprising of a number of single neurons have been presented of a definite number. But they consist of a different length of contexts. The statistical tagger annotates the input data based on the voting on the output of all single-neuron tagger. It provides eighty one percent of accuracy.[56]

[52] S. Dandapat, S. Sarkar and A. Basu, "*A Hybrid Model for Part-Of-Speech Tagging and its Application to Bengali*", In Proceedings of the International Journal of Information Technology, Volume 1, 5umber 4.,
[53] Ekbal Asif, et.al, "Bengali Part of Speech Tagging using Conditional Random Field" in Proceedings of the 7th International Symposium of Natural Language Processing (SNLP-2007), Pattaya, Thailand, 13-15 December 2007,
pp.131-136
[54] Ekbal, A. Bandyopadhyay, S., "Part of Speech Tagging in Bengali Using Support Vector Machine", ICIT- 08, IEEE International Conference on Information Technology, pp. 106-111, 2008
[55] Hammad Ali (2010), *An Unsupervised Parts-of-Speech Tagger for the Bangla language*", Department of Computer Science, University of British Columbia. 2010.
[56] Das, B. R., & Patnaik, S. (2014*), "A Novel Approach for Odia Part of Speech Tagging Using Artificial Neural Network"*. In Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2013, pp. 147-154. Springer International Publishing.

Recently there is a tagger developed using statistical approach by Pitamber Behara as his Mphil research. In this work a comparative study has been done that which approach gives better result between CRF++ and SVM. For training of the tagger corpora of 300k tokens or around 50k sentences has been used. The highest accuracy achieved by SVM was 96.85% .on the other hand CRF++ based tagger achieved 94.39% on seen data and 88.87 on unseen data.[57]

2.2.4. POS Taggers for Punjabi Language

Punjabi (or Panjabi) language is a member of the Indo-Aryan family of languages. It is also known as Indic languages. Other members of this family are Hindi, Bengali, Gujarati, and Marathi etc. Using rule based approach, a Panjabi POS tagger developed by Singh Mandeep, Lehal Gurpreet, and Sharma Shiv, in 2008. The system was developed as a sub part of grammar checker project. They have developed a fine–grained tagset, which contain around 630 tags. Using the rule based disambiguation approach a database was designed to store the rules. The performance of the system was manually correct and incorrect tag assignments and the system reports an accuracy of 80.29% including unknown words and 88.86% excluding unknown words.

The second POS tagger was developed by Sharma et al.(2011) .[58] This POS tagging system has been developed by using statistical method. Hidden Markov model was used to disambiguate the tags. For implementation of Hidden Markov model, Viterbi algorithm was used. In this work a hybrid approach which is a combination of rule based system and statistical approach was also experimented, in which the output of rule based system was fed to the statistical based system.

Hidden Markov Model (HMM) technique based tagger was developed by Manjit Kaur, Mehak Aggerwal, and Sanjeev Kumar Sharma in 2015.[59] The tegset of Singh Mandeep, Lehal, which has 630 tag reduced to 36 tags has been used by Technical Development of Indian Languages (TDIL) to improve the tagging accuracy of HMM based POS tagger. This improve tagger has shown an accuracy of 92-95%.

[57] Behera, P. (2015*). "Odia Parts of Speech Tagging Corpus: Suitability of Statistical Models. M. Phil. Dissertation"*. New Delhi: Jawaharlal Nehru University.
[58] Sharma S.K, Lehal G.S (2011) "*Using HMM to Improve accuracy of Punjabi POS tagger*" 2011 IEEE International Conference on computer science and Automation Engineering. Shanghai (China)
[59] Manjit Kaur, Mehak Aggerwal, and Sanjeev Kumar Sharma, (2015),*" Improving Punjabi Part of Speech Tagger by Using Reduced Tag Set*",International Journal of Computer Applications & Information Technology Vol. 7, Issue II Dec 14- January 2015 (ISSN: 2278-7720)

An HMM based POS tagger has been proposed in 2015 by Sirajuddin Y. Hala , and Sagar H. Virani. In this work the authors are using BIS Standard Tagset. The target data is 10,000 sentences for training purpose.[60]

2.3. POS Taggers for South Dravidian Languages

As stated by France (2014); “Dravidian languages are the family of around 70 languages spoken by more than 250 million people in primarily in South Asia, namely India, Pakistan and Sri-Lanka. The Dravidian languages are divided into South, South-Central, Central, and North groups; these groups are further organized into 24 subgroups. Indian constitution recognises four major literary languages namely Telugu, Malayalam, Tamil and Kannada. Dravidian language has a very rich morphological structure which is agglutinative”.

Some noticeable attempts were done in Dravidian languages like Tamil, Telugu, Malayalam and Kannada language. There are six different attempts for POS taggers developments in Tamil language. There are three different attempts in Telugu and two attempts in case of Malayalam. There is only one corpus based POS tagger was developed in Kannada language.

2.3.1. POS Taggers for Tamil

As similar to other Dravidian languages Tamil is also an agglutinative language. It is a verb final, relatively free-word order and morphologically rich language. According to Dhanalakshmi et al. (2009); “Tamil words are made up of lexical roots followed by one or more affixes. Because of this tagging a word in a language like Tamil is very complex. The main challenges in Tamil POS tagging are solving the complexity and ambiguity of words.”

A POS tagger was prepared by Ganesan based CIIL Corpus and tagset. An improvement over a rule based Morphological Analysis and POS Tagging in Tamil were developed by M. Selvam and A.M. Natarajan in 2009. Dhanalakshmi V, Anand Kumar, Shivapratap G, Soman KP and Rajendran S of AMRITA university, Coimbatore developed two POS taggers for Tamil using their own developed tagset in 2009.

Vasu Ranganathan developed a POS tagger for Tamil called “Tagtamil“[61] . This tagger was based on Lexical phonological approach. The index method was used to perform Morph-

[60] Sirajuddin Y. Hala 1 ,Sagar H. Virani2, (2015)“*Improve accuracy of Parts of Speech tagger for Gujarati language”,* International Journal of Advance Engineering and Research Development Volume 2,Issue 5, May -2015

tactics of morphological processing of verbs. The advantages of Tagtamil POS tagger is that, it handle both tagging and generation. Another Tamil POS tagger was proposed by Ganesan., which is based on CIIl corpus.[62] He used his own tagset, and he tagged a portion of CIIL corpus by using a dictionary as well as a morphological analyzer.

The third POS tagger system was proposed by Kathambam using heuristic rules based on Tamil linguistics for tagging, without using any dictionary or morphological analyzer. The system used twelve heuristic rules and then identifies the tags based on PNG, tense and case markers. There is a list of word which ahs been used by system to check stand alone words. Bigram approach has been used to tag unknown words.

M. Selvam and A.M. Natarajan in 2009 developed a tagger for Tamil using Projection and Induction techniques.[63] The tagger gives an improved rule based morphological analysis and POS Tagging in Tamil. The proposed idea was based on this purpose and achieved an improved accuracy of about 85.56%. Using an alignment-projection techniques and categorical information, a well organized POS tagged sentences in Tamil were obtained for the Bible corpus.

Dhanalakshmi V, Anand Kumar, Shivapratap G, Soman KP and Rajendran S of AMRITA University, Coimbatore developed a POS tagger for Tamil using Linear Programming approach.[64] They have developed their own POS tagset consists of 32 tags. A SVM methodology has been praposed, based on Linear Programming for implementing automatic Tamil POS tagger. For training of the system a corpus of 25000 sentences was used. The testing was performed using 10,000 sentences and overall accuracy of 95.63% was reported. In another attempt the same team has developed a POS tagger using machine learning techniques, where the linguistical knowledge is automatically extracted from the annotated corpus[65]. The tagset used to develop POS tagger was the same as earlier. This is a corpus based POS tagger. The annotated corpus size of 2250000 words was used for training (1,

---

[61] S. Rajendran (2006), "Parsing in Tamil", LANGUAGE IN INDIA www.languageinindia.com Volume 6: 8 August, 2006.

[62] Ganesan, M. (1994). "*Functions of Morphological Analyzer Developed at CIIL, Mysore*", in Harikumar Basi (ed.) Automatic Translation (seminar proceedings), Thiruvanthapuram: ISDL.

[63] M. Selvam, A.M. Natarajan (2009), *"Improvement of Rule Based Morphological Analysis and POS Tagging in Tamil Language via Projection and Induction Techniques*", International Journal of Computers, Issue 4, Volume 3, 2009.

[64] Dhanalakshmi V, Anand Kumar, Shivapratap G, Soman KP and Rajendran S (2009), *"Tamil POS Tagging using Linear Programming"*, International Journal of Recent Trends in Engineering, Vol. 1, No. 2, May 2009.

[65] Dhanalakshmi V, Anand kumar M1, Rajendran S, Soman K P., *"POS Tagger and Chunker for Tamil Language"*. Amrita Vishwa Vidyapeetham, Ettimadai, Coimbatore, Tamilnadu, India. Tamil University, Thanjavur, Tamilnadu, India.

65,000 words)) and testing (60,000 words). The algorithm used in development of this tagger was Support Vector Machine and the POS tagger system reported an accuracy of 95.64%.

2.3.2. POS Taggers for Telugu Language

Telugu is classified as a Dravidian language with heavy Indo-Aryan influence. Telugu language occupied 15th position in the world and 2nd position in India in the term of speakers. It is the official language of Andhra Pradesh. Telugu grammatical rule is deduced from a Sanskrit canon. Telugu uses many morphological processes to join words together, forming complex words.[66] Telugu is a highly inflectional and agglutinative language providing one of the richest and most challenging sets of linguistic and statistical features resulting in long and complex word forms. Telugu generally follows the Subject Object Verb (SOV) structure.

There are three noticeable POS tagger developments in Telugu, based on Rule-based, Transformation based learning and Maximum Entropy based approaches. An annotated corpus of 12000 words was constructed to train the transformation based learning and Maximum Entropy based POS tagger models.[67] The existing Telugu POS tagger accuracy was also improved by a voting algorithm by Rama Sree, R.J. and Kusuma Kumari P in 2007. The rule based approach uses various functional modules which works together to give tagged Telugu text. This system has different functional modules like Tokenizer, Morphological Analyzer, Morph-to-POS translator, POS disambiguator, unigram, bigram rules and Annotator. ambiguity is controlled by using unigram and bigram rules. Annotator is used to produce the tagged words in a text and reported accuracy of the system was 98%. The second attempt for development of a Telugu language POS tagger is based on Brill transformation rule based Learning (TBL). This POS tagger system consists of three phases of Brill tagger: Training, Verification and Testing. The reported accuracy of the proposed POS tagger is 90%. One Maximum Entropy approach based Telgu POS tagger was developed by R.J Ramasree, and P Kusuma Kumari. The proposed POS tagger was

[66] http://en.wikipedia.org/wiki/Telugu_language accessed (08.11.2015)
[67] Dinesh Kumar and Gurpreet Singh Josan,(2010), “Part of Speech Taggers for Morphologically Rich Indian Languages: A Survey”, International Journal of Computer Applications (0975 – 8887) Volume6–No.5, September, 2010.

implemented using publically available Maximum Entropy Modelling toolkit [MxEnTk] and the reported accuracy is 81.78%.[68]

2.3.3. POS Taggers for Malayalam

According to Dinesh and Josan (2010); "Malayalam is spoken primarily in southern coastal India having over 35 million speakers. Malayalam has its own distinct script which has a syllabic alphabet consisting of independent consonant and vowel graphemes plus diacritics. Malayalam belongs to the Dravidian family of languages and is one of the 22 constitutional languages of India. Malayalam has a rich literary tradition. Malayalam has an inflexional morphology. It heavily inflected by the addition of suffixes with the root/stem word. A rich amount of agglutination is also a major quality of Malayalam".

There are great deal of work have been done in the processing of Malayalam. A tagger for Malayalam had been proposed by Manju[69]. The tagger was based on HMM approach. A morphological analyzer was used for generating corpus as they did not have an annotated corpora. Then the corpus was used for training the HMM algorithm. The performance of the developed POS Tagger is about 90% and almost 80% of the sequences generated automatically for the test case were found correct.

Support Vector Machines (SVM) based tagger for Malayalam was developed by Antony P.J, Santhanu P Mohan and Dr. Soman K.P of AMRITA university Coimbatore. They used a SVMTool for tagging which was developed by Giménez and Màrquez[70]. Anthony et. al. first developed a tagset named AMRITA POS for Malayalam and used it for corpus annotation. A corpus size of about 180,000 tagged words were used for training the system their tagger reported 94% accuracy with their tagset.

2.3.4. POS Taggers for Kannada Language

Kannada language is both agglutinative and morphologically very rich. It is an ancient language which came into existence around 2000 years back, so it is declared as classical

[68] RamaSree, R.J, Kusuma Kumari, P., 2007 "Combining Pos Taggers For Improved Accuracy To Create Telugu Annotated Texts For Information Retrieval".
available at http://www.ulib.org/conference/2007/RamaSree.pdf
[69] Manju K., Soumya S., Sumam, M. I., (2009) "Development of a POS Tagger for Malayalam – An Experience". In: International Conference on Advances in Recent Technologies in Communication and Computing, pp.709-713.
[70] Jesus Giménez and Llúis Màrquez., (2006) "SVMTool. Technical manual v1.3".

language along with Sanskrit, Tamil, Telugu and Malayalam by Government of India. Approximately 6 crores of people speak Kannada around the world. It has its own script for reading and writing.

Antony P J and Soman KP of Amrita University, Coimbatore proposed statistical approach to build a POS tagger for Kannada language using SVM.[71] An accuracy of 86% is obtained in this work. They proposed a tagset consisting of 30 tags. The architecture of the proposed POS tagger in Kannada language, is based on corpus based and supervised machine learning approach. The Part-Of-Speech tagger for Kannada language was modelled using SVM kernel. A linguistic study to determine the internal linguistic structure of the Kannada sentence has been done. The tagset was developed based on that study. A corpus size of fifty four thousand words was used for training and testing the accuracy of the tagger generators. In 2011 Shiva et al have developed a set of cross language POS taggers for Kannada with Telugu resources.[72] The various models built are based on HMM model and show results comparable to existing mono-lingual POS taggers.

A Kannada POS tagger is proposed by Vijaylaxmi F. Patil and Shahid Mushtaq Bhat[73]. The corpus size for this work was 10,000 Kannada words from Aesthetics domain. The corpus was tagged according to BIS standards.

B R Shambhavi et. Al. (2010) has proposed a POS tagger for Kannnada Languages using Hidden Markov Model (HMM) and Conditional Random Fields (CRF).[74] Training data includes 51,269 words and test data consists of around 2932 tokens. EMILLE corpus was use for data collection . The accuracy of the tools based on HMM and CRF is 79.9% and 84.58% respectively.

[71] Antony P.J , Soman K.P. (2010) *"Kernel based Part of Speech Tagger for Kannada"* In Proceedings of the Ninth International Conference on Machine Learning and Cybernetics, Qingdao, pp 2139 – 2144. 11-14 July 2010

[72] Siva Reddy, Serge Sharoff. (2011) *"Cross Language POS Taggers (and other Tools) for Indian Languages: An Experiment with Kannada using Telugu Resources".* In Proceedings of IJCNLP workshop on Cross Lingual Information Access: Computational Linguistics and the Information Need of Multilingual Societies. Thailand, 2011

[73] Vijayalaxmi .F. Patil and Shahid Mushtaq Bhat, (2011). *"Part-ofSpeech Tagging for Kannada"* National Seminar on POS Annotation for Indian Languages: Issues and Perspectives. Organized by Linguistic Data Consortium for Indian Languages (LDC-IL), Government of India, 12-13 Dec 2011.

[74] Shambhavi B R, RamakanthKumar 2012. *" Kannada Part-Of-Speech Tagging with Probabilistic Classifiers".* International Journal of Computer Applications (0975 – 888) Volume 48– No.17, June 2012

2.4. POS Taggers for Resource-Poor Languages

This section of chapter deals with the description of POS Tagger developed for resource poor languages. Marathi, Gujrati, Magahi, Bhojpuri, Sambhalpuri, Assames, and Manipuri Taggers are discussed in this section.

2.4.1. POS Tagger for Marathi

Marathi is a morphologically rich language spoken by the native people of Maharashtra. H.B. Patil, A.S. Patil, B.V. Pawar proposed a Part-of-Speech Tagger for Marathi Language using Limited Training Corpora. It is also a rule based technique. They developed 25 SRR rule. Disambiguation is removed by the use of rule-base model or Hidden Markov Model. Based on the corpus they have identified 11 disambiguation rules that are used to remove the ambiguity. Stemming process removes all possible affixes, it change the meaning of stem word like (Anischit-Nischit).

Jyoti Singh Nisheeth Joshi Iti Mathur et.al has proposed a Marathi POS tagger using statistical techniques. They used statistical tagger using Unigram, Bigram, Trigram and HMM Methods. To achieve higher accuracy they use set of Hand coded rules, it include frequency and probability.

The approach used for development of tagger is statistical. In this approach Unigram, Bigram, Trigram and HMM Methods were used. A tag set for Marathi was also introduces which can be used for tagging Marathi text. The unigram tagger's accuracy is 77.38%, on the other hand Bigram and Trigram accuracy was 90.30% and 91.46% respectively. The highest 91.46% accuracy was achieved by HMM based tagger.[75]

2.4.2. POS Tagger for Gujarati

Gujarati is one of the less privileged languages in the sense of being resource poor. A POS tagger has been developed using Conditional Random Field approach, by Chirag Patel and Karthik Gali.[76] In this work, 600 sentences were manually tagged and this data has been used for training the tagger. The tagset used in this work is Indian Language (IL) tagset, which is a

[75] Jyoti Singh Nisheeth Joshi Iti Mathur, 2013," Development of Marathi Part of Speech Tagger Using Statistical Approach" , Advances in Computing, Communications and Informatics (ICACCI),.
[76] Chirag Patel, Karthik Gali (2008) "*Part-Of-Speech Tagging for Gujarati Using Conditional Random Fields*". In Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages, pages 117–122,

standard tagset. There are 26 different tags in this tagset. For training the system both tagged (600) sentences and untagged (5000) sentences are used. This tagger has achieved an accuracy of 92% for Gujarati texts. The tagger used training corpus of 10,000 words. The test corpus was of 5,000 words.

2.4.3. POS Tagger for Magahi

Magahi is one of the smaller Indo-Aryan languages. It is spoken in Eastern state of Bihar in India. Magahi is a resource-poor language. Kumar Ritesh (e.l) has developed a tagger for Magahi. For this work four taggers were used – first one is Support Vector Machines (SVM) based SVMTool. The second tagger is Hidden Markov Model (HMM) based TnT tagger. Third tagger of Magahi is Maximum Entropy based MxPost tagger. Fourth tagger is based on Memory based MBT tagger. The dataset used for training these taggers a dataset of 50,000 words has been used and tested on 13,000 words. The tagset used in the training data is BIS-tagset for Indian languages. Highest accuracy is achieved by Maximum Entropy Tagger (89.61%). The performance was tested against a frequency-based baseline tagger.[77]

2.4.4 POS Tagger for Marathi[78]

In 2014, Pallavi Bagul, et.al. have developed a rule based pos tagger for Marathi language. In this tagger, assigns part of speech to the words in a sentence given as an input. Authors used a corpus which is based on tourism domain called annotated corpus and three grammar rules are used for the experiment to resolve ambiguous word which acts a noun and adjective in certain context, or act as an adjective and adverb in certain context.[79] Another tagger was proposed by Jyoti Singh, et.al. (2013) , for tagging Marathi using Trigram method.[80]

[77] Kumar Ritesh, Lahiri Bornini, Deepak Alok, (2014*), " Developing a POS tagger for Magahi : A comprehensive study"*.

[78] Pallavi Bagul, Archana Mishra, Prachi Mahajan, Medinee Kulkarni, Gauri Dhopavkar, *"Rule Based POS Tagger for Marathi Text"* in proceeding of: International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 1322-1326.

[79] Rathod Shubhangi, Sharvari Govilkar, (2015), *"Survey of various POS tagging techniques for Indian regional languages",* / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (3) , , 2525-2529

[80] Jyoti Singh Nisheeth Joshi Iti Mathur, (2013) *"PART OF SPEECH TAGGING OF MARATHI TEXT USING TRIGRAM METHOD"* International Journal of Advanced Information Technology (IJAIT) Vol. 3, No.2.

2.4.5 POS tagger for Manipuri

Manipuri (Meiteilon) is one of the oldest languages in the South-East Asia. It has its own script (Meitei Mayek) and literature. Presently it is used to write in Bengali Script. One rule based tagger has been developed by Department of Computer Science, Assam University, Silchar. This is a lexicon based tagger. This works has a tagset consisting 97 tags with generic attributes and language specific attribute value. The tagger has a Tokenizer, to separate the words, a Stemmer which separates the affix i.e.; prefix and suffix from the root words, Splitter, Morphological Analyzer and a Lexicon. The dataset used for this work was from tourism domain. Highest accuracy of the tagger is 92% on 1500 token after applying 35 rules.[81]

2.4.6 POS Tagger for Assamese

Assamese is a morphologically rich and relatively free word order and agglutinative language like any other Indian language. Using HMM model with Viterbi algorithm; a tagger of Assamese language has been developed by Navanatha Saharia. The tagger achieved an accuracy level of 87%.

Pallav Dutta (2013) has attempted to develop an online semi-automated tagger. This was designed to deal with sparse data problem of the language. NLTK is used to tag the test data and for the ambiguous tags an online tagger would help to change the tags[82].

2.4.7 POS Tagger for Sambalpuri

Sambalpuri is a Eastern Indo-Aryan language, which is low-density lesser-known, poorly-described, less-resourced, minority or less-computerized language. It has fewer resources available. Pitambar Behara, Atul Ojha and Girish Nath Jha have developed a POS tagger for Sambalpuri[83]. The POS tagger was developed with the application of SVM and CRF++. A corpus of approximately 121k was collected from the web and annotated using the BIS

[81] Singha, Kh Raju, Ksh Krishna Bati Singha, Bipul Syam Purkayastha,(2013), "*Developing a Part of Speech Tagger for Manipuri",* International Journal of Computational Linguistics and Natural Language Processing, Vol 2 Issue 9 September 2013 ISSN 2279 – 0756

[82] Dutta Pallav Kumar,(2013) *"An online Semi Automated Part of Speech Tagging Technique applied To Assamese",* Thesis submitted to IIT Guwahati India.

[83] Behera Pitambar , Atul Kr. Ojha and Girish Nath Jha,*"Issues and Challenges in Developing Statistical POS Taggers for Sambalpuri"*Centre for Linguistics, Jawaharlal Nehru University.

(Bureau of Indian Standards) annotation scheme devised for Odia under the ILCI (Indian Languages Corpora Initiative) Corpora Project[84]. Both the taggers are trained and tested with approximately 80k and 13k respectively. The SVM tagger provides 83% accuracy while the CRF++ has 71.56% which is less in comparison to the former.

2.4.8 POS Tagger for Sindhi

Javed Ahmed MAHAR, Ghulam Qadir MEMON proposed a system for "Rule Based Part of Speech Tagging of Sindhi Language".[85] This tagger take input text, and generate token. Once token generated search then it compare selected word from lexicon (SWL) . If word is found one or more times, then store associated tag and if not found add that word into lexicon by generating linguistic rule for new word. The tagset contains 67 tags. A lexicon named SWL is developed having entries of 26366 words. A set of 186 disambiguation rules are used for SPOS tagging system. The contextual information is used for rule-based approach and manually assigns a part of speech tag to a word. Accuracy of 96.28% was achieved from SPOS. The increase in tagged data and rules can be used for increasing the accuracy of the tagger.

2.4.9 POS tagger of Bhojpuri

The POS tagger for Bhojpuri has been developed by a group of linguists at JNU. The tagger has been developed using SVM approach. The tagger has been trained on the data 90K tokens with an accuracy of 82% to 89 %. The data was from general domain and collected using a web crawler.[86]

2.5. POS Taggers of Sanskrit

This section of the chapter deals with a description related with Taggers developed in Sanskrit. There are very less work done in this area.

[84] http://sanskrit.jnu.ac.in/projects/ilci.jsp?proj=ilci

[85] Javed Ahmed MAHAR Ghulam Qadir MEMON,(2010)*"Rule Based Part of Speech Tagging of Sindhi Language "* proceeding of International Conference on Signal Acquisition and Processing.

[86] Singh, Srishti. (2015). "*Challenges in Automatic POS Tagging of Indian Languages- A Comparative Study of Hindi and Bhojpuri".* M. Phil. Dissertation. New Delhi: Jawaharlal Nehru University.

2.5.1. Sanskrit rule-based tagger

A rule based tagger has been developed in Jawaharlal Nehru University as a part of Ph.D research of R. Chandrashekhar (2006). A stable POS tagset for Sanskrit text evolved, which has 65 word class tags, 43 feature sub-tags, and 25 punctuation tags and one tag **UN** to tag unknown words – a total of 134 tags. As part of the research, various tagged linguistic resources were developed. Tagged noun and *avyaya* lexicons were developed from Digital Monier Williams Sanskrit English Dictionary and other sources. Also a tagged *tiṅanta* lexicon and tagged *subanta* example base were prepared.

2.5.2 Sanskrit Tagger

This tagger has been developed by Oliver Hellwig. SanskritTagger is a stochastic tagger for un-pre-processed Sanskrit text. The tagger tokenises text with a Markov model and performs part-of-speech tagging with a Hidden Markov model. Parameters for these processes are estimated from a manually annotated corpus of currently about 1.500.000 words.[87] It is a freeware under a permissive license and standalone application.

2.5.3. Tree Tagger for Sanskrit

Namrata Tapaswi, Suresh Jain [88] proposed a Treebank Based Deep Grammar Acquisition and Part-Of-Speech Tagging for Sanskrit Sentences. For experimental result Author taken set of 100 words and manually evaluated, the system gives 90% correct tags for each word. The evaluation was done in two stages. Firstly by applying the lexical rules and secondly, after applying the context sensitive rule. The highest accuracy achieved by tagger was 69.2 %.

[87] http://www.indsenz.com/int/index.php?content=sanskrit_tagger

[88] Namrata Tapaswi Suresh Jain , *"Treebank Based Deep Grammar Acquisition and Part-Of-Speech Tagging for Sanskrit Sentences"* Software Engineering (CONSEG), 2012 CSI Sixth International Conference on.

CHAPTER 3:

POS TAGGING IN SANSKRIT LANGUAGE

Introduction-

This chapter discusses the features of Sanskrit language. The chapter gives an insight into the grammar of Sanskrit. In this process P*āṇ*ini, his work Aṣṭādhyāyī (AD) and its components have been discussed. Further the chapter deals with Sanskrit morphology and gives a details of its categories. Computation and Sanskrit and sentence formation are also discussed in this part. The third part of the chapter gives a detail about the history of Part of Speech (POS) in Sanskrit. At last the chapter discusses the problems of Sanskrit which create obstacles for developing language technology in Sanskrit.

3.1. Sanskrit

Sanskrit (*saṁskṛta* 'adorned, purified [by grammar]'; Cardona 1997:557-64) is a form of Old Indo-Aryan used over a wide area in the north of the Indian subcontinent from around the middle of the second millennium BC onwards. Starting from Vedas, the language has a history of an enlightening intellectual tradition and extensive literature. To this day, it remains a vehicle for original literature as well as technical works. Sanskrit is officially recognized in the eighth schedule of the Indian Constitution and has also been reported by a lot of speakers as their mother tongue in the census of India.[89]

Morphology is the branch of linguistics that deals with words. It is concerned with how words are formed or created in a language from smaller units in a systematic way. It has to find as well as describe the mechanism behind this process. Each language has morphemes which are its smallest meaningful units. Realization of morphemes as part of a word is called morph. In formation of words from these smaller units, certain processes such as inflection, derivation and compounding etc. are involved. Also involved is morphotactics that determines how morphs should be put together to form words.

3.1.1. System of P*āṇ*ini

P*āṇ*ini's grammar AD (approximately 7th BCE) is important for linguistic computation for two reasons. One, it provides a comprehensive and rule based account of a natural language in about 4000 rules - the only complete grammatical account of any language so far. Two, the

[89] George Cardona and Dhanesh Jain (eds.), "***The Indo-Aryan languages***", Routledge Language Family Series, vol. 2, London and New York: Routledge.

model of a 'grammar-in-motion' that it provides seems to closely mimic a fully functional Natural Language Processing (NLP) system[90]. P*āṇ*ini's grammar AD (approximately 7th BCE) is important for linguistic computation for two reasons. One, it provides a comprehensive and rule based account of a natural language in about 4000 rules - the only complete grammatical account of any language so far. Two, the model of a 'grammar-in-motion' that it provides seems to closely mimic a fully functional Natural Language Processing (NLP) system[91]

The Sanskrit grammar is based on the spoken language (bhāṣā) of Pāṇinī's time. It also gives rules on Vedic usage. The regional variants also get benefited with Pāṇinī's work. Its optional (apavāda) rules distinguish between preferable and marginal forms.Some of the rules even have sociolinguistic conditions. It is entirely synchronic: variants are simply treated as alternate forms; indeed, the very concept of linguistic change is foreign to the tradition. The grammar consists of four components:

3.1.1.1. Aṣṭādhyāyī (AD)

The rules of the AD make reference to classes defined on the elements in the other three components by means of conventions spelled out in the AD itself. Thus, while none of the components is intelligible in isolation, together they constitute a complete integrated system of grammar.

There are also various peripheral adjuncts to the system. The most important of these are the *uṇādigaṇa*, which extend the Pāṇinīain technique to analyze unproductive and irregularly formed derivatives from roots. Though mentioned in a few rules of the AD , many of the words they derive are treated as underived there, and they are probably post- Pāṇinīain at least in their present form.[92]

A*ṣṭā*dhy*ā*y*ī* (7th BCE) is a composite text including the following components -

1. *akṣara-samāmnāya* or *māheśvara-sūtra* (14) (AS)
2. *śabdānuśāsana* or *sūtrapāṭha* (3965 or 3983 in *kāśikāvṛtti* (SP)
3. *dhātupāṭha* (1967 verb roots - 2014 including *kaṇḍvādi* roots) (DP)

[90] http://www.languageinindia.com/feb2004/panini.html
[91] ibid
[92] Paul Kiparsky, Paninian Linguistics. *Encyclopedia of Languages and Linguistics*, 1993

4. *gaṇapāṭha* (other pertinent items like primitive nominal bases, *avyayas*) (GP)

Sūtrapāṭha **(SP)** is arranged in eight *adhyāyas* (chapters) each divided into four sub-chapters (*pādas*). The SP has approximately 3965 rules (*sūtras*) which have been arranged in x.x.x format (to be accessed in as *adhyāya . pāda . sūtra* format)[93]

AD has mainly six type of sūtra.[94]. These are -

1. Saṃjñā
2. Paribhāṣā
3. Vidhi
4. Niyama
5. Atideśa
6. Adhikāra

The following is a summary of topics discussed in the AD[95] -

First chapter-This chapter serves as an introduction to the work this book contains. This chapter deals with the major definitional and interpretational rules named saṃjñāsūtrāṇi rules i.e *adeṅ guṇaḥ*. The indicators (*it*) (1.3.2-1.3.9), related rules are the main topic of discussion and discussed in details.There are rules related with extension *(atideśa).* rules dealing with *atmanepada-parasmaipada* (1.3.12-1.3.93) and rules dealing with the *kāraka* (1.4.23-1.4.55) definitions and *nipāta* (1.4.56-1.4.97) are given in this chapter.

Second Chapter -This chapter has rules dealing with compounds (*samāsa*) (2.1.3-2.2.38). All the compounds named *avyayībhāvaḥ, tatpuruṣaḥ, bahuvrīhiḥ, dvandvaḥ* related rules are given here. This chapter has rules related with nominal inflection. Assignment of cases (2.3.1-2.1.73) is given with the rules of replacement relative to roots. Rules dealing with number and gender of compounds (2.4.1-2.4.31) are another component of this chapter. This chapter also deals with rules related with deletation by *luk* (2.4.58-2.4.84).

---

[93] Jha Girish Nath 'The System of Panini' Language in India, volume 4:2 February 2004
[94] *saṃjñā ca paribhāṣā ca vidhir niyama eva ca*
atideśa adhikāraśca ṣaḍavidhi sūtra lakṣaṇam
[95] Sharma, Rama Nath, The A*ṣṭā*dhy*āyī* of P*ā*ṇini – Volume-I page-75-76

Third Chapter- This chapter contains rules related with suffix. All the suffixes which connect with verb, are given here. In this these suffix rules dealing with derivational of roots ending in affixes *sanādayaḥ*. Those which are rules related with the derivational of ending in a *kṛt* are given. *vikaraṇa* related rules , lakāra related rules are the further given. The derivational of ending in a *tiṅ* related rules are also given here

Fourth Chapter -This chapter contains rules dealing with derivation of a *pada* ending in a *sup*. Those rules which deal with feminine affixes (*strī*) are given here. All the rules dealing with the derivational of nominal stems ending in an affix termed *taddhita* are also the part of this chapter.

Fifth Chapter -This chapter has rules related with taddhitapratyayāḥ and samāsāntādhikāraḥ. In taddhitapratyayāḥ topic there are two main sub-topic vibhaktisaṃjñakāḥ pratyay and avyayasaṃjñakāḥ pratyay.

Sixth Chapter The sixth chapter has rules dealing with doubling (dvitva).rules dealing with *samprasāraṇa* and *saṁhitā.* Rules dealing with the are given here. The other topic of rules are augment (*āgama*) *suṭ* and accent. The last *Pada* deals with phonological operations relatives to a pre-suffix base (*aṅga*)

Seventh chapter This chapter has a wide set of rules related with pre-suffix base and rules dealing with operations relative to affixes augment etc. *Iḍāgama* and *vṛddhiprakaraṇa* related rules are the part of this chapter.

Eighth Chapter-In this chapter the saṃdhi related rules are given.

With this description on can say that first five chapters are give the basis to build an infrastructure, to give suitable information to execute the derivation easily. The last three chapter deals with the rules which carry out the changes related with a continuous text, word accent and the shape of stem. Kapoor(1992) has classified the content of A*ṣṭā*dhy*ā*y*ī* into four parts[96].According to him the first and second chapters deals with classification and enumeration of bases and categories, the third and fourth chapters contains of *prakṛti-pratyaya* enumeration, and derivation of bases, The sixth chapter to eight chapter's first part deal with the synthesis of *prakṛti-pratyaya*. Last 3 parts of chapter eight deals with the rules of morphophonemic.

[96] Kapoor, Kapil, "Text Interpretation: The Indian Tradition"

3.1.1.2. The Components of AD

AD is a very précised and well-structured text. Its subject matter has those properties which makes it close to a formal grammar. Some of the prominent components are discussed here :-

Śiva s*ū*tras or Praty*ā*h*ā*ra S*ū*tra (PS)

The purpose of the PS component is to enumerate all Sanskrit phonemes. P*āṇ*ini has formed fourteen *sūtras* or sets with shared phonetic attributes and has ordered the sets in the order of decreasing vocalicity. Each *sūtra* ends with an indicatory letter/sound called IT (marked by capitals in the table) which is not a member of the set to which it applies the closure. These meta-linguistic markers, also called *anubandhas*, are used in generating *pratyāhāras*/ 'set-denoters'/ siglas by applying a well defined method (*ādirantyena sahetā*). A sigla is formed by taking one letter/sound and one of the IT sounds and denotes all the phonemes in between except the ITs. Thus, *al* denotes to the list of all phonemes, *ac* refers to all vowels, *hal* to all consonants and *ñam* to all nasals. This technique has enabled P*āṇ*ini to economically specify the domain of application of various rules without having to enumerate every time all the member phonemes of that set. The 14 sutras are-

Sūtra	**Devan*ā*gar*ī***	**Class of sounds**
a i u Ṇ	अ इ उ ण	Simple vowels
ṛ ḷ k	ऋ ऌक्	'Sonant' vowels
e o *ṅ*	ए ओङ्	Dipthongs
ai au c	ऐ औच्	Dipthongs
h y v r *ṭ*	ह य व रट्	Voiced aspirate + semi-vowels
l *ṇ*	लण्	Semi-vowel *l*
ñ m *ṅ ṇ* n m	ञ म ङ ण नम्	Nasal stops
jh bh *ñ*	झ भञ्	Palatal and bilabial voiced aspirates
gh *ḍ*h dh *ṣ*	घ ढ धष्	Voiced aspirates stops
j b g *ḍ* d ś	ज ब ग ड दश्	Voiced unaspirated stops
kh ph ch *ṭ*h th c *ṭ* t v	ख फ छ ठ थ च ट तव्	Unvoiced aspirated stops followed by unaspirated stops
k p y	क पय्	Unvoiced unaspirated stops
ś *ṣ* s r	श ष स र	Sibilants
h l	हल्	Voiced fricative

Table 3.1.   *Māheśvara sūtra*

In principle, hundreds *pratyāhāras* could be formed from these *sūtras*. P*ā*ṅini has used 43 (of a 44th introduced by later grammarians, *rañ*=(*r*,*l*))[97]. In these, some of the *prtyāhāras* are ambiguous.

For example, *ṇ* occurs twice in the list, so it can be assigned with two different meanings by assigning to *pratyāhāra aṇ*[98] including or excluding *ṛ* etc.

[97] ङ्णटञ्वात् स्मृतो हृयेकः, चत्वारश्च चमान्मताः ।
शलाभ्यां षड् यरात्पञ्च, षाद् द्वौ च कणतस्त्रयः ॥
केषाञ्चिच्च मते रोऽपि, प्रत्याहारोऽपरो मतः ।
लस्थाऽवर्णेन   वाञ्छन्त्यनुनासिकबलादिः ॥

[98] अणुदित्सवर्णस्य चाप्रत्ययः

***Sūtrapāṭha* (SP)**[99] The main body of the description is *Sūtrapāṭha* or the list of sūtras/ aphorisms composed by P*ā*ṇini in order to describe the language. The SP contains about 3965 *sūtras* or 3983 in *kāśikā vṛtti* arranged in chapters (*adhyāya*) and sub-chapters (*pāda*) in a particular order[100].

Chapter	**P*ā*da I**	**P*ā*da II**	**P*ā*da III**	**P*ā*da IV**	**Total Rules**
1st	74	73	93	109	349
2nd	71	38	73	85	267
3rd	150	188	176	117	631
4th	176	144	166	144	630
5th	135	140	119	160	554
6th	217	198	138	175	728
7th	103	118	119	97	437
8th	74	108	119	68	369
Total Rules in A*ṣṭā*dhy*ā*y*ī*					**3965**

Table 3.2: Distribution of AD *sūtras*

Sūtras are defined as "brief but unambiguous, concise but comprehensive, impersonal and objective"[101] verb-less sentences. Besides that, some of the chief features of *Sūtra*-style are:

- Careful avoidance of repetition of the same term in a string.
- Use of certain technical terms which are defined in the introduction and are pertinent only for the particular subject.
- Use of technical devices which help in making out the sense
- Abstraction from a series of observations through categorization.

Thus *sūtras* are the formula or program that gives a an impression like code. They can be categorized in following types[102] –

[99] Chandra Subhash, 2006, Mchine Recognition And Morphological Analysis Of Subanta-Pada, Dissertation submitted to Jawaharlal Nehru University, new delhi
[100] Shastri, Bheemsen, Laghusiddaantkaumudi Ist part, page: 5
[101] *alpākṣaramasandigdhaṃ sāravadviśvatomukham|*
astobhamanavadyañca sūtraṃ sūtravido viduḥ ||

- ***Samjñā* (Technical Rules):** Rules which assign a particular term to a given entity.These are the technical rules. Ex. *adeṅ guṇāḥ*
- ***Paribhāṣā* (Interpretive Rules):** In this category all the rules which regulate proper interpretation of a given rule or its application are included.
- ***Vidhi* (Operational Rules):** Those rules, which are used to perform when an operation has given on basis of a given input.
- ***Niyama* (Restriction Rules):** All the rules which restrict the scope of a other rules are known as *Niyama*.
- ***Atideśa* (Extensions Rules):** In this category those rules are included which expand the scope of a given rules. They usually do this by allowing the transfer of certain properties which were otherwise not available.
- ***Adhikāra* (Heading Rules):** These types of Rules are those which introduce a domain of rule that shares a common topic, operation, input, physical arrangement, etc.

Dhātupāṭha* (1967 verb roots - 2014 including *kaṇḍvādi* roots) (DP)[103]-** The lexicon of Sanskrit verb roots is named as *dhātupāṭha*. SP component is used for assuming or explicitly calling it. Including *kaṇḍvādi* roots; there are 1967 verb roots in P*ā*ṅini *dhātupāṭha.** It is organized into ten classes as follows –

Sr.	***Class***	***Total roots***	***Modification***
1	Bhvādi	1035	*ś*ap
2	Adādi	71	luk
3	Juhotyādi	24	*ś*u
4	Divādi	141	*ś*yan
5	Svādi	34	*ś*nu
6	Tudādi	155	*ś*a
7	Rudhādi	25	*ś*nam
8	Tanādi	10	u
9	Kryādi	62	*ś*nā
10	Curādi	410	*ś*ap
Total	**10**	**1967**	**10**

Table 3.3. Distribution of DP

[102] संज्ञा च परिभाषा च विधिर्नियम एव च ।

अतिदेशोऽधिकारश्च षड्विधं सूत्रमुच्यते ॥

[103] Chandra Subhash, 2006, Mchine Recognition And Morphological Analysis Of Subanta-Pada, Dissertation submitted to Jawaharlal Nehru University, new delhi

Ga*ṇ*ap*āṭ*ha (GP)

In GP, primitive nominal bases are listed in groups. In GP, each group (*gaṇā)* used for a similar function. The bases are the various clases like *sup, tiṅ, taddhita, strī, kṛt.* Eighteen *upasargas* with 23 pronouns, operates on these nominal bases.The example of these *gaṇā are sarvādi, ajādi, śaradādi* etc.[104]

3.2. Sanskrit morphology[105]

The study of the structure and form of words in languages or a language, including inflection, derivation, and the formation of compounds is called morphology. In Sanskrit, a syntactic unit is called *pada*. **Cordona[106] (1988)** posits the formula for Sanskrit sentence **(N-E^n)p…(V-E^v)p.** In Sasnkrit *Pada* can be nominal (*subanta*) or verbal (*tiṅanta*). These forms of *Pada* are formed by inflecting the stems and hence they are part of Sanskrit inflectional morphology. The derivational morphology in Sanskrit studies primary forms (*kṛdanta*) and secondary forms (*taddhitānta*), compounds (*samāsa*), feminine forms (*strī pratyayānta*) etc.

P*āṇ*ini defines *pada* as '*sup-tiṅ-antam padam*'[107] – (bases) that end with either *sup*-suffixes or *tiṅ*-suffixes. The *pada*-s with *sup*-suffixes are called *subanta-pada*-s and the *pada*-s having *tiṅ*-suffixes are called *tiṅanta-pada*-s (constituting NPs and VPs respectively in a sentence). According to P*āṇ*ini the indeclinables: *upasarga*-s, *karmapravacanīya*-s, adverbs are all *subanta-pada*-s with zero *sup*-suffixation.

As stated earlier that; Sanskrit has two types of morphology- nominal and verbal. There are approximately 2014 verb roots (including *kandvādi),* classified in 10 *gaṇas*.In Sanskrit the derived verb forms can have 12 derivational suffixes[108]. These can have *ātmanepadi* and *prasmaipadi*. A verb root may have approximately 2190 (tense, aspect, number etc.) morphological forms. Sanskrit Nominal morphology of two types, Primary [*kṛdanta* (roots forms that end with *kṛt* suffixes)] and secondary [*taddhitānta* (noun forms that end with

[104] Jha Girish Nath 'The System of Panini' Language in india http://www.languageinindia.com/feb2004/panini.html access on 20/10/2015
[105] Chandra Subhash, 2006, Mchine Recognition And Morphological Analysis Of Subanta-Pada, Dissertation submitted to Jawaharlal Nehru University, new delhi
[106] **George Cardona, 1988** Pãnini, His Work and its Traditions, vol **...** i (Delhi: **MLBD**, **1988**)
[107] सुप्तिङन्तं पदम् । A*ṣṭ*. I.4.14
[108] सन्–क्यच्–काम्यच्–क्यङ्–क्यषोऽथाऽऽचारक्विब्–णिज्यङस्तथा ।

यगाय ईयङ् णिङ् चेति द्वादशाऽमी सनादयः ॥

taddhita suffixes)]. Secondary nominal morphology may be of following types like-*samāsānta* (compound nouns), *strīpratyayānta* (feminine forms) etc. They can also include *upasargas* (prefix) and *avyayas* (indeclinables) etc. According to Pāṇini, there are 21 morphological suffixes (seven *vibhaktis* and combination of three numbers = 21)[109] which are attached to the nominal bases (*prātipadika*)[110] according to syntactic category, gender and end character of the base.

3.2.1. Inflectional morphology

Inflectional morphology in Sanskrit is of two types: Nominal inflectional or declensional (*subanta*) morphology and Verbal inflectional or conjugational (*tiṅanta*) morphology.

3.2.1.1. Nominal inflectional morphology (subanta)[111]

The nominal base is called as *prātipadika.* It is defined as 'any meaningful linguistic unit (*arthavat*) which is neither a verbal root (*adhātu*) nor an affix (*apratyaya*)'.[112] To the *pratipadika*-s, 21 nominal suffixes called as *sup-pratyaya*-s[113] (7 cases in three numbers) get inflected to form a *subanta-pada*-s. All non-verb categories are *subanta-padas* in a Sanskrit sentence. It is necessary to analyse *Subanta-padas,* before stating any type of computer processing. In Sanskrit, *subanta* forms are can be potentially very complex.There are four types of *subanta* forms. These are primary (*kṛdanta*) and secondary (*taddhitānta*), feminine forms (*strīpratyayānta)* and compound nouns (*samāsa).* Some time, *upasargas* and *avyayas* can also get included. There are 21 morphological suffixes, named as *vibhaktis.* With 21 morpholgical suffix these *vibhaktis's* forms a combination.[114] These attach to the nominal bases (*prātipadika*) according to the syntactic category of the base, gender and end character of the base. According to Pāṇini; *sup* suffixes are ***su, au, jas,am, auṭ śas, ṭā, bhyām, bhis, ṅe, bhyām, bhyas, ṅasi, bhyām, bhyas, ṅas, os, ām, ṅi, os, sup***. These suffixes are in the sets of three as- (***su, au, jas***) (***am, auṭ, śas***) **(*ṭā, bhyām, bhis*)** (***ṅe, bhyām, bhyas***) (***ṅasi, bhyām,***

[109] स्वौजसमौट्छष्टाभ्याम्भिस्ङेभ्याम्भ्यस्ङसिभ्याम्भ्यस्ङसोसांङ्योस्सुप ।४।१।२॥
[110] अर्थवधातुरप्रत्ययः प्रातिपदिकम् ।१।२।४५॥, कृत्तद्धितसमासाश्च ।१।२।४६॥

[111] Chandra Subhash, 2006, Mchine Recognition And Morphological Analysis Of Subanta-Pada, Dissertation submitted to Jawaharlal Nehru University, new delhi
[112] अर्थवदधातुरप्रत्ययः प्रातिपदिकम् । A*ṣṭ*. I.2.45
[113] स्वौजसमौट्छष्टाभ्याम्भिस्ङेभ्याम्भ्यङसिभ्याम्भ्यस्ङसोसाम्ङ्योस्सुप् | A*ṣṭ*. IV.1.2
[114] स्वौजसमौट्छष्टाभ्याम्भिस्ङेभ्याम्भ्यस्ङसिभ्याम्भ्यस्ङसोसांङ्योस्सुप्

bhyas) (***ṅas, os, ām***) (***ṅi, os, sup***)[115] for singular, dual and plural[116] respectively. To obtain inflected forms (*subanta padas*); these suffixes are added to the *prātipadikas*[117]. The definition of prātipadikas is any meaningful form of a word that is neither a root nor a suffix). These *prātipadikas* can be categorized in two types: primitive and derived. The primitive bases are those that are stored in g*aṇapāṭha*, which is a collection of bases with similar forms. On the other hand, derived *prātipadikas* are formed by adding the derivational suffixes. They are the donators of unity, duality and plurality. There are Some words, which always comes in the singular. For example *ekaḥ*(one). There are some that are always dual in nature like *dvi* (two), *akṣi* (eyes) etc. while some of them are always plural like *apaḥ* (water), *dārāḥ* (wife) etc. *Subanta* can be categorized is mainly six types –

3.2.1.2. *Avyaya subanta* (Indeclinable)

Avyaya subanta-padas are indeclinable. They remain unchanged under all morphological conditions[118]. According to P*āṇ*ini [2.2.82][119], if affix feminine suffixes e.g. *cāp*, *ṭāp*, *ḍāp*, and *sup* come after *avyaya*, then they get deleted by *luk*. According to P*āṇ*ini *avyayas* are defined as *svarādinipātamavyayam* [1.1.36], *kṛnmejantaḥ* [1.1.38], *ktvā tosun kasunaḥ* [1.139] and *avyayībhāvaśca* [1.1.40][120] etc.

3.2.1.3. Nominal *Subanta* (Base) According to Chandra (2006) “Nominal *subantas* are basic *subanta*, which are *prātipadika* by *arthavadadhāturapratyayaḥ prātipadiakam”*. For example: *rāmaḥ, śyāmaḥ, pustakālayaḥ, vidyālayaḥ* etc.

3.2.1.4. *Samāsānta Subanta* (compound NPs) As stated by Chandra (2006); “When any Simple words (*padas*), whether substantives, adjectives, verbs or indeclinables, is added with another *subanta pada* Then it forms *samāsa* (compound)”. In Sanskrit Compound (*samāsas)* are categorized into four categories. These are; *avyayībhāva* (Adverbial), *tatpuruṣa*

---

115 सुपः

116 द्व्येकयोर्द्विवचनैकवचने

117 अर्थवधातुरप्रत्ययः प्रातिपदिकम् ।१।२।४५॥, कृत्तद्धितसमासाश्च ।१।२।४६॥

118 सदृशं त्रिषु लिङ्गेषु सर्वासु च विभक्तिषु ।
वचनेषु च सर्वेषु यन्न व्येति तदव्ययम् ॥ [गोपथ ब्राह्मण]

119 अव्ययादाप्सुपः [२.४.८२]

120 स्वरादिनिपातमव्ययम् [१.१.३६], कृन्मेजन्तः [१.१.३८], क्त्वा–तोसुन्–कसुनः [१.१.३९], अव्ययीभावश्च [१.१.४०]

*(*Determinative), *Bahuvrīhi (*Attributive), *dvandva* (Copulative). *Dvandva* and *Tatpuruṣa* compounds can be further divided into sub-categories.

3.2.1.5. ***Kṛdanta subanta*** (primary derived NPs)

In Sanskrit; The primary derivatives are known as *kṛdanta*. According to Chandra (2006); ”The primary affixes are to be added to verbs to derive substantives, adjectives or indeclinable *kṛt*. For example *paṭhitavyam, pātavya, paṭhanīya, pacelima, jeyam, deyam, karttā, kumbhakāraḥ, janamejayaḥ, pāṭhakaḥ, paṭhantī, gantum, khāditum, svapnam gatiḥ, gatvā,vihāya, ādāya* etc”.

3.2.1.6. *Taddhitānta subanta* (secondary derived NPs) In Sasnkrit morphology, the secondary derivative affixes are called *taddhita.* These *taddhita*, derive secondary nouns from *prātipadikas.* For example - *dāśarathī, gauṇa* etc. In his work Pāṇini has listed many *taddhita* suffixes. In these *taddhita* suffixes some are- *a, akañc, ac, añ, aṇ, at, iṣṭhan, īyasun, kan, ḍhak, ḍhañ, tamap, tarap, tayap, tal, tyap, tral, dvayasac, fak, matup, mātrac, yak, yat, yañ, ḍāc, kha, gha, cha, uraca.*

3.2.1.7. Tiṅanta (Verb inflectional morphology)

Verb base is called as *dhātu*. *dhātu*-s can be primary and derived. Primary *dhātu*-s are listed in the Dhātu-paṭha of Aṣṭādhyāyī. There are about 2000 verb roots in the list. Verb (*tiṅanta*) morphology in Sanskrit is very complex. Verbs carry information about tense (*kāla*), aspect - mood (*artha*), person (*puruṣa*) and voice (*vācya*). Apart from this verbs get inflected with *tiṅ-pratyaya*-s in three persons (*puruṣa-traya*) and three numbers (*vacana-traya*). In Sanskrit, verbs do not carry gender information. Due to the complexity of *tiṅanta*s, some of the forms have less usage (like desideratives), rare (like frequentatives) and few no more employed in their proper sense (distinction in the *parasmai* and *ātmane* usages, distinction in the of aspect in *laṅ, liṭ, luṅ*). The Vedic *leṭ* has totally become obsolete.[121]

Sanskrit verb forms are very complex. They held information of tense, aspect, number all in the inflection forms. There are about 2000 verb roots in Sanskrit, which are classified in 10 morphological and semantic classes. They are further classified in *ātmanepadī* and *parasmaipadī* forms in 10 *lakāra* and 3 x 3 persons and numbers combinations. They can also be potentially. 12 secondary suffixes are there. When these suffix are added to verb roots,then

[121] Spiejer, J.S., (1886), ‘Sanskrit Syntax’, Reprint 1998, Motilal Banarsidas, Delhi – 07, p.228

they create new verb roots. Mishra & Jha (2004) have done a rough calculation of all potential verb forms in Sanskrit to be more than 10,29,60,000.

3.2.2.1. Derivational morphology

In derivational morphology, the meaning of words changes by applying derivations. A combination of word stem with a morpheme is called a derivation, which forms a new word. This word is often of a different class, for example, *gam* becomes *gantṛ, gantum* and *gatavat.* The suffixes *tṛc, tumun* and *ktavatu* change the class and meaning of the base word. Derivational Morphology can be quite complicated.

In Sanskrit nominal bases (*prātipadika)* can be primary or derived. Primary *prātipadika*-s are stored in the Ga*ṇ*a-p*āṭ*ha of A*ṣṭā*dhy*ā*y*ī.* There are four kinds of nominal derivations: *kṛt* (primary affixation to the verb root), *taddita* (secondary affixation to the verb root), *samāsa* (compounding) and *strīpratyaya* (feminine affixation). The derived *prātipadika*-s are the *kṛtanta*, *taddhitānta*, *samāsa* and the *strī-pratyayānta*-s.

3.2.2.1. Kṛdanta

kṛdanta-s are the primary nominal derivatives from the verb roots. The derived *kṛdanta*-s fall into categories of substantives, participles, gerunds, infinitives and indeclinables, all of them fall into the main category *subanta*-s according to P*āṇ*ini.

The derived *kṛdanta* fall into categories of *substantives*, *participles*, *gerunds, infinitives* and *indeclinables*. when a verb root is operated with *kṛt* suffixes then the derived form is *kṛdanta*. According to Bha*ṭṭ*ojid*ī*k*ṣ*ita in *Siddhāntakaumudī*; *kṛt suffixes* are of three kinds

3.2.2.1.1. *kṛtya* suffixes

kṛtya suffixes are those that are used in *bhāva-vācya* and *karma-vācya*. They are in neuter singular.Examples of *kṛtya* suffixes are; *tavyat, tavya, anīyar, kelimar, yat, kyap, ṇyat etc. paṭhitavyam, pātavya, paṭhanīya, pacelima, jeyam, deyam,* etc are also called *kṛtya* suffixes.

3.2.2.1.2. *Pūrvakṛdanta* suffixes

Pūrvakṛdanta suffixes are those which are always used in *kartṛvācya* only.The example of *Pūrvakṛdanta* suffix are; *aṇ, ṭa, khaś, khac, kvanip,ṇvul, tṛc, lyu, ṇini, ac, śānac, śākan, u, kvip, itra, ka, ḍa, kta, ktavatu, śatṛ, śānac, śākan, u, kvip, itra, etc. Karttā, kumbhakāraḥ, janamejayaḥ, pāṭhakaḥ, paṭhantī* are the example words which has been derived by these.

3.2.2.1.3. *Uttarakṛdanta* suffixes

These suffixes are *tumun, ghañ, erac, ap, ktṛ, athuc, nañ, nan, ktin, khal, yuc, ktvā, lyap, ṇamula, uṇa etc.* For example*, gantum, khāditum, svapnam, gatiḥ, gatvā, vihāya, ādāya* etc.

3.2.2.2. Taddhit*ā*nta

The secondary nominal derivatives from the the primary *prātipadika*-s or derived *pratipadika*-s are *taddhitānta*-s. The *taddhita* affixes are added in various senses like – of showing possession (*matvarthīya*), - the son/daughter/descendent of (*apatyārtha*), - abstract noun (*imanic*), adverbs (*akac, tasi* etc.)[122]

3.2.2.3. Str*ī*-pratyay*ā*nta

Sanskrit has eight feminine suffixes *ṭāp, cāp ḍāp, ṅīs, ṅīn ṅīp, uṅ* and *ti* and words ending in these suffixes are called *strīpratyayānta*. For example - *ajā, gaurī, mūṣikā, indrāṇī, gopī, aṣṭādhyāyī, kurucarī, yuvatī, karabhorū* etc. *strīpratyayānta* forms inflected with *sup* are called *strīpratyayānta-subanta*.

Feminine *prātipadika*-s are derived from the masculine *prātipadika*-s by the addition of affixes *ā* (*ṭāp, ḍāp, cāp*), *ī* (*ṅīp, ṅis, ṅit*), *ū* (*ūṅ*) and *ti.*[123]

3.2.3. Sam*ā*sa

[122] Kale, M.R, (1995), 'A Higher Sanskrit Grammar', MLBD Publishers, New Delhi, pp.194-216.
[123] Ibid. p180.

In Sanskrit, a *pada* - substantives or adjectives or indiclinables (also verbs) can combine with another *subanta-pada* to form compounds (*samāsa*). They are principally classified into four types: (a) *dvandva* or Copulative, (b) *tatpuruṣa* or Determinative, (c) *bahuvrīhi* or Attributive, and (d) *avyayībhāva* or Adverbial.[124]

Simple words (*padas*), whether substantives, adjectives, verbs or indeclinable, added with another *subanta-padas* are called *samāsa* (compound). Sanskrit *samāsas* are divided into four categories, some of which are in turn divided into sub-categories. The four main categories of compounds are as follows: (1) adverbial or *avyayībhāva*, (2) determinative or *tatpuruṣa*, (3) attributive or *bahuvrīhi* and (4) copulative or *dvandva*. *dvandva* and *tatpuruṣa* compounds can further be subdivided into sub-categories.[125]

For any natural language processing system for Sanskrit one has to process the derived nouns but it is difficult to identify the derived nouns in a sentence. Typically the identification of derived nouns will start after the *prātipadika* has been separated from the *sup* suffixes. Thereafter the *prātipadika* has to be identified as one of the four types- *kṛdanta, taddhitānta, strīpratyayānta* and *samāsānta*. Without doing it we can not meaningfully process or understand Sanskrit text for example-

(1) *Jagadguruḥ, Gopālakaḥ*

After Subanta processing it can be rewritten as-

(2) *Jagadguru + su [pum; pra; ek], Gopalaka + su [pum; pra; ek]*

Now the problem is to interpret the derived nouns in (2) as *kṛdanta, taddhitānta, strīpratyayānta* and *samāsānta.*

3.2.4. Kāraka and vākya

A*ṣṭā*dhy*ā*y*ī* does not as such discuss the concept of sentence (*vākya*), as its primary motive is to generate complete syntactic words called *pada*-s. But there are other non-technical words

[124] Ibid, pp.113-115

[125] Chandra Subhash, 2006, Mchine Recognition And Morphological Analysis Of Subanta-Pada, Dissertation submitted to Jawaharlal Nehru University, new delhi

like ‘*kāraka*’, ‘*samānādhikaraṇa*’, ‘*samartha*’, ‘*anabhihita*’, ‘*sākāṅkṣa*’ etc. used in A*ṣṭā*dhy*ā*y*ī* to express relationships between *pada*-s.

kāraka rules are those which explain a situation in terms of action (*kriyā*) and its factors (*kāraka*) are those which have a function in the fulfilment of any action (*kriyā-anvayitvaṃ kārakatvam*). The semantic relationship that hold between the nouns and verbs in a sentence is specified by *kāraka*-s. These *kāraka*-s are expressed thorough *vibhakti*-s. The six *kāraka*-s are: 1. *apādāna* (Source), 2. *sampradāna* (Beneficiary) 3. *karaṇa* (Means), 4. *adhikaraṇa* (Locus), 5. *karman* (patient), 6. *kartṛ* (agent).[126]

Later grammarians have defined the sentence variously. Pata*ñ*jali discusses this in his Mah*ā*bh*āṣ*ya. After various discussions he defines sentence as that ‘which has verb along with attributes’[127] where he includes adverbs, adjectives and *kāraka*-s within attributes (*viśeṣaṇa*).

3.3. Component of Sanskrit sentence

In Sanskrit a sentence consist *subanta padas* (NPs) and *tiṅanta padas* (VPs). According to **Cordona[128] (1988)** a sentence can be defined in following formula :-

$$(\mathbf{N - E^n})\mathbf{p} \ldots (\mathbf{V - E^v})\mathbf{p}$$

In this formula first of all *sup* and *tiṅ* combines with *prātipadikas*. After this process they get assigned with syntactico-semantic relation through *kāraka*. This stipulation makes complete sentences.

3.4. Sanskrit and Computation

Sanskrit is virtually an unexplored treasure for modern computational linguistics and AI researchers. The Sanskrit language was studied to a high degree of formalization from high

[126] Jha, Girish Nath, (1993), ‘Morphology of Sanskrit Case Affixes: A Computational Analysis’, M.Phil dissertation submitted to JNU, pp.17-24.

[127] आख्यातं सविशेषणम् वाक्यम् । M.Bh. II.1.1.7

[128] **George Cardona, 1988** Pāṇini, His Work and its Traditions, vol **...** i (Delhi: **MLBD**, **1988**)

antiquity by genius linguists such as P*āṇ*ini, and a continuous tradition of commenting and refining his work (K*ā*ty*ā*yana, Pata*ñ*jali, Bhart*ṛ*hari, N*ā*ge*ś*a Bha*ṭṭ*a etc.), still very much alive, leaves hope for the emergence of new computational models of linguistic treatment, well tuned by definition to the Sanskrit language. The Sanskrit corpus contains a wealth of knowledge and wisdom (most of which is also available in electronic media) which has not been yet properly brought to light, despite centuries of both Western and Indian scholarship. Sanskrit benefits from meticulously checked databases of verb forms, noun paradigm classes, and a host of other information necessary for development of computational tools that can be used for analysis and generation of Sanskrit texts.[129] Because of these reasons Sanskrit can very well be the 'lingua franca' for NLP research.

In the case of Sanskrit computation there are some theoretical issues to be kept in mind. The first thing is the difference in the nature of Natural language (NL) and Artificial language (AL) and the status of Sanskrit. To understand the need and possibilities of formalisation of Sanskrit language and grammar, the advantages and disadvantages of NL and AL are to be kept in mind. On one hand, AL is necessary for communicating with machine, on the other, NL is more expressive and capability of handling human communication needs. NL is more ambiguous and AL is less ambiguous. NL tolerates errors, but AL has zero tolerance for errors. NL is used in Human to Human communication while AL is used in Human to Machine and Machine to Machine communication. It appears that Sanskrit stads somewhere between a NL and an AL – more precise than a typical NL and less mechanical and more expressive than an AL.

The second issue is the nature of Sanskrit language. In comparison to other languages, Sanskrit is easily adaptable for computing because it has a regular structure, smaller speech community with almost nonexistent regional variations. . P*āṇ*ini's grammar is the only complete grammar for any natural language so far. It has been used as a NL for communicating in day to day life and also as an adapted metalanguage for writing precise sh*ā*straic expressions.

The third point which goes in favour of Sanskrit is the fact that most NL applications need fair amount of knowledge computing for robustness and usefulness purposes. Sanskrit contains explict theories of knowledge in *Nyāya* and *Mīmāṁsā Śāstra* and also a continuous lexicographic tradition

[129] Jha Girish et.al., 2007, Proc. Of FISSCL, INRIA, Paris, Pg. No. iii

While Sanskrit can help NLP on various fronts, it can also be helped in following ways:

- Electronic storage and publication
- Access and preservation of Sanskrit corpus- Vedic/Laukika
- Sanskrit computational lexicography
- Sanskrit-Indian languages Machine Translation
- Automatic reading of manuscripts (OCR)
- Textual interpretation
- Computational pedagogy of Sanskrit
- Building interactive indices, glossary etc.

Sanskrit Language Processing has a great significance in the context of Indian Language Processing. The reason behind this argument is that Sanskrit has great influence on other languages. Because of Its effect Sanskrit is known as the mother to all other Indian Language.

The reason which makes Sanskrit very important in linguistic arena is

1) Sanskrit has a well formed grammar formalism.
2) It has less variability. Sanskrit has not change much from last 2500 years.
3) Any natural language tool prepared on the basis of Sanskrit grammar rule can be well-suited on the other Indian language, with a few changes.
4) Annotated corpus of Sanskrit is beneficial tool for the researchers of NLP, speech recognition and other AI areas.

It proves that Sanskrit could be a basic building block for constructing statistical models, that can be used for automatic processing of natural languages.

3.5. Brief History of POS

In the grammar, POS (also known as lexical category, grammatical category, lexical class and word class) is a linguistic category of words that are categorised either by their morphological or/and syntactic behaviours. POS is also defined as the logical distinct category of word-classes dealing with the grammatical facts. Classification of words into POS is different from the grammatical categorization which is based on morphological structure of words. Grammatical category is the categorization of lexical items on the basis of their intrinsic properties. The main grammatical categories are noun, adjective, verb, and adpositions. Morphological structure of words refers to the morpho-phonemic forms conveying the notion of gender, number, person, tense, mood, case-ending etc which are basically inflections. Joshi defines POS as a 'logical categorization presenting the general procedure for the classification of words upon a plan which, although supported by logic, is in no way contrary to the grammatical facts'.[130]

The famous eight way classification of POS which is seen in English grammar is taken from Graeco-Roman grammar which classifies words into eight categories viz. noun, verb, participle, pronoun, preposition, adverb, article and conjunction (adjective and interjection are also seen in addition to this list). More recent lists of POS for English language have very large numbers of word classes, for example in the Penn Treebank[131] there are 45 categories. There are 87 word classes of English are in the Brown corpus[132], while C7 tagset has 146 categories[133].

POS can be broadly categorized into two: closed class and open class. Among the POS categories like pronouns, prepositions are called closed class POS as these have fixed membership. Nouns and verbs categories are referred as open class because in these categories new nouns and verbs can be added or borrowed from other languages. Generally, Closed class words are function words. Function words include those words which occur very frequently in a language and are very essential basic ones like personal pronouns, demonstrative pronouns, conjuncts, etc.[134]

[130] Joshi, S.D., (1966), 'Adjectives and Substantives as Single Class in Parts of Speech', in the Journal of University of Poona, Vol.25, 19-30.
[131] Penn Treebank Tagset - http://www.computing.dcu.ie/~acahill/tagset.html (accessed: 25.11.2015)
[132] Brown Corpus Tagset - http://www.comp.leeds.ac.uk/amalgam/tagsets/brown.html (accessed: 25.11.2015)
[133] C7 Tagset - http://www.comp.lancs.ac.uk/ucrel/claws7tags.html (accessed: 25.11.2015)
[134] Jurafsky, Danier & Martin, James H., 2002, Speech and Language Processing, Pearson Education Inc., New Delhi, p.289.

3.6. History of POS in Sanskrit or *Padajāta*

Sanskrit language is highly inflectional and morphologically very rich. Morphological criteria are the necessity for deciding the membership in lexical categories in Sanskrit. Nouns and adjectives with verbs form one category having inflections while the adpositions form a different category without any inflections. Again nouns and adjectives take declensional inflections (*sup* suffixes) and verbs take conjugational inflections (*tiṅ* suffixes). Nouns have intrinsic gender and adjectives have gender in accordance to the noun. This seems to be a distinguishing feature among the two, but there are many exceptions, thus making the distinction between the two very difficult.[135]

Sanskrit has a long tradition of linguistic thought beginning with the Veda-s. Grammar (*vyākaraṇa*), phonetics and phonology (*śikṣā*), study of etymology (*nirukta*) and metrics (*chandas*) along with astrology (*jyautiṣa*) and ritual science (*kalpa*) are the six auxiliaries to the Veda directly concerned with linguists.[136] Among these grammar is considered as the main auxiliary.[137] There has been a long tradition of grammarians, linguists and etymologists in the history of Sanskrit literature. Pada-p*āṭ*ha, the word-by-word the analysis of the Vedic text and Pr*ā*ti*śā*khya-s, the grammars relating to a particular branch of Veda (*śākhā*) are the early and primary works studying words and their classification.

Y*ā*ska

The foremost among the Sanskrit grammarians is Y*ā*ska. In his Nirukta, the commentary on the Nigha*ṇṭ*u - a lexicon of Vedic words, Y*ā*ska classifies[138] POS, what he calls *padajāta*[139] into four: *nāma* (noun), *ākhyāta* (verb), *upasarga* (preverb), and *nipāta* (particle). Semantically *nāma* and *ākhyāta* are to distinguished categories. According to Y*ā*ska, *ākhyāta* denotes action with a definite sequence of beginning, middle and end. while *nāma* is those words which denotes frozen action or substance but not a process. He explains *upasarga* or

[135] Dash, Sinuruddha, (1995), The Syntax and Semantics of Sanskrit Nominal Compounds, University of Madras, Madras, p.30-31

[136] शिक्षा व्याकरणं छन्दो निरुक्तं ज्यौतिषं तथा। कल्पश्चेति षडङ्गानि वेदस्याहुर्मनीषिणः॥

[137] प्रधानं च षट्स्वङ्गेषु व्याकरणम्।M.Bh. Paspa*ś*a

[138] चत्वारि पदजातानि नामाख्याते चोपसर्गनिपाताश्च। Nirukta I.1

[139] पदजातानि पदगणा इत्यर्थः। जातशब्दो हि गणे प्रसिद्धः। तद्यथा गोजातमश्वजातमिति। तद्वदिहापि। Durga on Nirukta I.1

prepostions as words which bring into prominence the subordinate meaning of nouns and verbs. He subdivides particles into comparatives, conjunctives, and explectives and explains in detail all the particles with examples.

Durg*ā*c*ā*rya

Durg*ā*c*ā*rya, is the famous commentator on Nirukta. He gives account of the different ancient schools and teachers of grammar having different view on the number and definition of POS:

- The Aindra school of grammar accepts word as a single category. According to themd the word is that which has sense.
- Some schools, whom Durg*ā*c*ā*rya, hasn't given name, consider two classes of words – *subanta* and *tiṅanta*
- Another school of grammar considered three classes of words – *subanta*, *tiṅanta* with *upasarga* and *nipāta* as a combined third class.
- Y*ā*ska and others classify into four.
- Some other schools added *gati* and *karmapravacanīya* to the Y*ā*ska's classification and making the numbers as five or six[140]

P*āṇ*ini

P*āṇ*ini in his great monumental work A*ṣ*t*ā*dhy*ā*y*ī* classifies *pada*[141] based on their morphologically. According P*āṇ*ini words can be categories into two broad categories: words having nominal inflections (*sup-anta*) and words having verbal inflections (*tiṅ-anta*). To become a word, the root needs an inflection. The nominal stem which receives nominal inflections is called as *prātipadika*. *prātipadika* can be defined as a meaningful unit which is neither a root nor a suffix[142]. The *prātipadika* can be [either primitive (listed in *gaṇapāṭha*) or] *kṛt* (primary) derivations, *taddhita* (secondary) derivations and *samasa* (compounding)

[140] नैकं पदजातं यथार्थः पदमैन्द्राणामिति। नापि द्वे यथा सुबन्तं तिङन्तं च। नापि त्रीणि निपातोपसर्गौ एकतः कृत्वा। नापि पञ्च षड्वा गतिकर्मप्रवचनीयभेदेनेति। Durga on Nirukta I.1

[141] सुप्तिङन्तं पदम्। A*ṣṭ*. 1.4.14

[142] अर्थवदधातुरप्रत्ययः प्रातिपदिकम्। A*ṣṭ*. 1.2.45

formations[143]. Panini includes particles, prepositions, adjectives, adverbs also together as *prātipadika*. He has also used nouns and adjectives as different categories. This is suggested by his usage of terms *viśeṣaṇa* and *viśeṣya*.[144] He also uses the word *guṇavacana* in the sense of adjective while prescribing the *iṣṭhan* and *īyas* suffixes which he makes an injunction only to be used with *guṇavacana*-s.[145] He also defines and classifies the particles (*nipāta*) - verbal prepositions (*upasarga & gati*) and verb like usages (*karmapravacanīya*).[146] But he categorizes particles, prepositions, adverbs, pronouns and adjectives under the noun (*subanta*) category.[147]

Patañjali

Patañjali is a brilliant scholar and grammarian who has written commentary on Aṣṭādhyāyī. Patañjali acknowledges the four way classification of Yāska several times in his 'Great Commentary' Mahābhāṣya on the Pāṇini's work. He defines sentence (*vākya*) as 'verb (*ākhyāta*) along with an indeclinable (*sāvyaya*), syntacto-semantic categories (*sakāraka*), and adjectives (*saviśeṣaṇa*), during his commenting on the Kātyāyana's Vārttika[148]. He adds adverb (*sakriyāviśeṣana*) to the list and refines the definition as 'sentence is verb with its modifiers', by including all the other categories into modifier (*saviśeṣaṇa*) as they all act as modifiers to the verb.[149]

Patañjali gives another classification of words. This classification is based on semantic. The words are classified according to what they signify. For example those word which suggests general classes (*jāti-śabdāḥ*), words signifying properties (*guṇa-śabdāḥ*), words telling actions (*kriyā-śabdāḥ)* and those that signify individuals (*yadrcchā-śabdāḥ*). The *jāti-śabdas*

[143] कृत्तद्धितसमासाश्च। A*ṣṭ*. 1.2.46

[144] विशेषणं विशेष्येण बहुलम्। A*ṣṭ*. 2.1.57

[145] अजादी गुणवचनादेव। Ast. 5.3.58

[146] उपसर्गाः क्रियायोगे। Ast 1.4.59 कर्मप्रवचनीयाः। Ast. 1.4.83

[147] अर्थवदधातुरप्रत्ययः प्रातिपदिकम्। Ast. 1.2.45; स्वरादिनिपातमव्ययम्। Ast. 1.1.37; अव्ययादाप्सुपः। Ast. 2.4.82; स्वौजसमौट्छष्टाभ्याम्भिस्ङेभ्याम्भ्यस्ङसिभ्याम्भ्यस्ङसोसाम्ङ्योस्सुप्। Ast. 4.1.2

[148] आख्यातं साव्ययकारकविशेषणं वाक्यम्। Kātyāyana wrote Vārttika-s, the critical annotations on Pāṇini's sutra-s to cover the language of his time. Vārttika is defined as 'the exposition of the meaning, of that which is said, of that which is left unsaid, and of that which is ill or imperfectly'. Patañjali comments on these Vārttika-s, and thus Vārttika-s have become a part of Mahābhāṣya.

[149] M.Bh. 2.1.1.7

are common nouns, *guṇa-śabda-s* are adjectives, *kriyā-śabda-s* are verbs and *yadṛcchā-śabda-s* are proper nouns.[150]

Bhartṛhari

Bhartṛhari is a great grammarian and philosopher who has composed *Vākyapadīya*. *Vākyapadīya* is the treatise on comprehension of the meanings of words and sentences. Although it is a grammatical work; but it also has a profound philosophical background. According to Bhartṛhari, grammar is a school of philosophy. Bhartṛhari has discussed about the classification of the words in the third chapter of *Vākyapadīya,* while beginning the discussion on the general or universal class in the *jātisamuddeśa* section. In this part Bhartṛhari mentions the view point of different grammarians on the classification of words. According to Bhartṛhari, sentence is the basic linguistic unit and words and their meanings are obtained by artificial division (*apoddhāra*) of the sentence. This analysis and classification of these artificial units of the sentence are done variously by different grammarians. Three ways of classification by different grammarians has been mentioned in *Vākyapadīya*.[151] Some grammarians classify words into two (*dvidhā*) categories: verb or action (*ākhyāta*) and noun or accessory to action (n*ā*man).These scholar include particles (*nipāta*), prepositions (*upasarga*) and postpositions (*karmapravacanīya*) under nouns and verbs. Some other grammarians have classified words into four categories. The counts *nipāta* and *upasarga* as different categories along with the *nāma* and *ākhyāta*. In the third way they take *karmapravacanīya* as separate with the rest four making it a 5 way classification (*pañcadhā*).[152]

3.7. Sanskrit POS Problems

There are several serious issues coming in the way of building a POS tagger for Sanskrit. In Sanskrit **ambiguity** is the key issue. While designing a POS tagger, it is necessary to address ambiguity and solved it. Sandhi, homonyms, lack of punctuation marks, and implicit

[150] चतुष्टयी शब्दानां प्रवृतिः। जातिशब्दाः गुणशब्दाः क्रियाशब्दाः यद्दच्छाशब्दाश्चतुर्थाः। M.Bh. I.19.10

[151] द्विधा कैश्चित् पदं भिन्नं चतुर्धा पञ्चधापि वा। अपोद्धृत्यैव वाक्येभ्यः प्रकृतिप्रत्ययादिवत्॥
Vp.III.Jātisamuddeśa Kā.1

[152] Iyer, K.A. Subramania, (1971), The Vakyapadiya of Bhartrhari, Ch. III, pt. I, English Translation, Deccan College, Pune

knowledge are some issues. Words behave different, according to their context. Therefore; identifying a POS tag of a token appearing in a particular context, is a challange.

These problems are discussed in following points:-

3.7.1. High level of Ambiguity [153]

According to Joshi[154], *Bhartṛhari* determines the categorization of *viśeṣaṇa* and *viśeṣya* semantically and not morphologically.[155] Another word used for adjective is *viśeṣaṇa* which literally means a qualifier. A *viśeṣaka* can be a qualifier of either a noun or a verb. So here the *viśeṣaṇa* category includes adjectives and adverbs together. The difference between these two categories is that the adverb, being an action qualifier, does not decline and remain similar in all genders, cases and in all numbers.[156] But the adjective, *nāma- viśeṣaṇa*, does decline in all genders, cases and numbers and follows the gender, number and case as that of the noun.[157] But Cardona observes that there is no particular provision made in the P*ā*ṇini's grammar for concord between adjectives and nouns qualified by them. He cites from *Bhāṣāvṛtti* of Puruṣottamadeva and shows that the adjective, having the property of the noun, being in the same locus (*samānādhikaraṇa*) as that of the noun takes the case and gender of the noun.[158]

Eg. *evaṃ* <u>*samatikrāmatsu*</u> <u>*keṣucit*</u> <u>*divaseṣu*</u> *rājā candrāpīdasya yauvarājyābhiṣekaṃ cikīrṣuh pratīhārān upakaraṇa-sambhāra-saṇgrahārtham ādideśa.*[159]

<u>*nivṛttena*</u> <u>*hṛdayena*</u> <u>*katicana*</u> <u>*kṣaṇān*</u> *pratīkṣasva.*[160]

It is very difficult to categorise adjectives (*viśeṣaṇa*) and substantives (*viśeṣya*) in Sanskrit. K*ā*ty*ā*yana and Pata*ñ*jali [161] point out that adjectives can be turned into substantives and

[153] Chandrashekar, R.: Parts-of-Speech Tagging For Sanskrit. Ph.D. thesis submittedto JNU, New Delhi (2007)
[154] Joshi, S.D., 1966, 'Adjectives and Substantives as a Single Class in the Part-of-Speech', Journal of the University of Poona, Vol.25, pp. 25
[155] विशेषणविशेष्यत्वं पदयोरुपजायते। न प्रातिपदिकार्थश्च तत्रैवं व्यतिरिच्यते॥ Vp. III 706
[156] सदृशं त्रिषु लिङ्गेषु सर्वासु च विभक्तिषु। वचनेषु च सर्वेषु यन्न व्येति तदव्ययम्॥ M.Bh. on Ast. I.1.38
[157] यल्लिङ्गं यद्वचनं या च विभक्तिर्विशेष्यस्य। तल्लिङ्गं तद्वचनं सैव विभक्तिर्विशेषणस्यापि॥ Samasacakra, p.28.
[158] Cardona, George, 1973, 'Indian grammarians on adverbs', *Issues in Linguistics*: papers in honour of Henry and Renee Kahane, ed. by Braj Kachru et. al., Urbana, p. 93.
[159] K*ā*dambar*ī*, *Ś*ukan*ā*sopade*ś*a
[160] B*ā*lar*ā*m*ā*ya*ṇ*a, Ara*ṇ*yak*āṇ*da p.34

substantives into adjectives. The view point of the speaker is the indicator of the Difference between *viśeṣya* and *viśeṣaṇa*. Because of that these can get interchanged.

Pata*ñ*jali tries to solve this problem when he used '*guṇa-vacana*' for adjective. According to him '*guṇa-vacana*' are the denotative of the qualities. On the other hand substantives are '*dravya-vacana*' that denotatives of substances.[162]

The only difference between substantive and adjective is that the former has intrinsic gender while the later has no intrinsic gender and follows the gender of the substantive which it modifies. Even this grammatical distinction is not always borne in mind. Eg. *śataṃ brāhmaṇāḥ*. Here the quantifying adjective *śata* does not concord with the substantive *brāhmaṇa* in gender and number. Distinction between a *dravya* and *guṇa* and the philosophical discussion, or the dual nature of pronouns is out of scope of this research endeavour.

Helar*ā*ja, the commentator on Bhartrhari's V*ā*kyapad*ī*ya, while commenting on Vp.III.1.3[163] says that pronouns (*sarvādi*) are subclass of nouns (*vastumātrābhidhāyin*) or adjectives (*viśiṣṭavastuvācaka*).[164] According to Bhartrhari, pronouns function either as adjectives or as nouns.

This discussion was just to highlight that the categories like adjectives and substantives and pronouns and also numerals in Sanskrit are dilute and have problems of ambiguity

---

[161] विशेषणविशेष्ययोः उभयविशेषणत्वात् उभयोः च विशेष्यत्वात् उपसर्जनाप्रसिद्धिः। Varttika, M.Bh. I.399.4

[162] यत्र हि अन्यतरत् द्रव्यम् अन्यतरः गुणः तत्र यत् द्रव्यं तत् प्रधानम्। तत् यथा शुक्लम् आलभेत कृष्णम् आलभेत इति न पिष्टपिण्डीम् आलभेत कृती भवति। अवश्यं तद्गुणं द्रव्यम् आकाङ्क्षति। Ibid.

[163] वस्तूपलक्षणं यत्र सर्वनाम प्रयुज्यते। द्रव्यमित्युच्यते सोऽर्थो भेद्यत्वेन विवक्षितः॥Vp. III.1.3

[164] इह सर्वनाम्नां द्वयी गतिः। वस्तुमात्राभिधायिनः केचिद्यथा सर्वादयः। विशिष्टवस्तुवाचकाश्चान्ये यथान्यतरादयः। Helaraja Commentary on Vp. III.1.3

3.7.2. Saṃdhi Formation

The euphonic rules of Sanskrit which are called saṃdhi are the obstacle in the processing of the language. These are the regular phonological transformation in Sanskrit. Although these can be solved by using saṃdhi splitter and pre-processing; still they create a lot of confusion both for human and machine.

3.7.3.Homonyms

Sanskrit is morphologically and lexically very rich language. Use of a great number of homonyms is a common phenomenon. This feature of Sanskrit makes it very ambiguous for automata. For example word kesara, appears with the four basic meanings. These are ”mane”, ”(lotus) fibre”, ”(a plant name) Mesua ferrea L.” and name of a mountain”.

3.7.4. Lack of Punctuation marks

Sanskrit lacks in punctuation marks. Only daṃḍas are used in narrative texts. In Sasnkrit texts Daṃḍa are indication of the end of a (complex) narrative substructure. Except that Sanskrit texts do not use any kind of reliable punctuation. In metrical texts daṃḍa are used for marking the end of a verse. This happans often, but by far not regularly, occurs with the end of a syntactic structure. For example this phenomenon appears in the end of subordinate clause. It proves that although daṃḍas are helpful in determining the hypotheses about the syntactic structure of text. But they do not behave as punctuation marks in a strict sense. The lack of punctuation marks creates an ambiguous environment which makes any tagging or parsing process very difficult.

3.7.5. Need of implicit knowledge

Sanskrit has very rich morphological and lexical features. These features make it hard to process for both human and machine. An expert trained in traditional knowledge is required to resolve the ambiguity, which is quite expensive, time taking and labour intensive

These feature need to take care during the computation of Sanskrit , as resolving them makes the works of the system easy and quite efficient.

CHAPTER 4:

SANSKRIT TAGSET AND PREPARATION OF THE TAGGED DATA

Introduction

This chapter discusses the topics related to tagset design and tagging of corpora. In this chapter, problems related to deciding a particular tagset and its preparation related issues are discussed. Also the detailed information about the tagsets designed for Indian languages. The prominent tagset like IIT (IL) Tagset, Microsoft research India Limited (MSRI), Indian Languages Consortium (LDC) and Bureau of Indian Standard (BIS) ; are some tagset whose features have been discussed here. The reason behind the selection of the Bureau of Indian Standard (BIS) and rationale behind its use for tagging of data is also given. The last section of the chapter deals with the preparation of linguistic resources. In this part the problem of annotating Sanskrit, the stages of annotation in Sanskrit, process of corpora creation, validation and tokenisation are discussed.

4.1. Tag set

Tagset is the set of tags from which the tagger is supposed to choose to attach to the relevant word.[165] Every tagger will be given a standard tagset. A tagset consists of tags that are used to represent the grammatical information of the language.

Different languages have different kinds of features. Because of this, the tagset designing and selecting tagset for tagging a language becomes a difficult job for the researcher. Any tagset can have as many tags as much information one wants to encode for that language.

For Indian languages several POS tagsets have been designed by a number of research groups working on Indian Languages viz IL-POST scheme, ILMT tagset, BIS tagset, AUKBC tagset, Tamil tagset, JNU-Sanskrit tagset (JPOS), Sanskrit consortium tagset (CPOS), MSRI Sanskrit tagset (IL-POST) and CIIL Mysore tagset.

4.1.1. Classification of Tagsets

There are two kind of classification of the tagsets based on the methods of using tags. These are :-

- Structure
- Depth of information

[165] http://language.worldofcomputing.net/pos-tagging/parts-of-speech-tagging.html

a) **Tagsets according to the information depth**

- **Coarse-grained Tagset-**

This category of tagsets has basic word categories related information, such as N (Noun), V(Verb), ADJ(Adjective), ADV(Adverb), PREP (Preposition), CONJ(Conjunction).

- **Fine-grained tagset-**

In the fine grained tagset, the word class gives deeper information such as N_NN(Noun-Nominative), N_NNP (Noun-Proper Noun), V_VM_VF(Verb-Main Verb- Finite Verb), and so on. According to (Hardie, 2004), "A POS tagset design should take into consideration all possible morphosyntactic categories that can occur in a particular language or group of languages."

b) **Tagsets according to the tag structure**

- **Flat Tagset**

In the flat tagsets there is no clear structural distinction between major word class information and extra feature information. They have large number of tags, with their separate labels. It is not possible to scale any structure from these tags. But they are easy to build. The famous C5 Tagset used for annotating the BNC Corpus is flat tagset.

- **Hierarchical Tagsets**

In this kind of tagsets contain less number of tags compared to flat tagset. They are modular, related structurally, easily scalable, requires more research to develop, and can be compared to related object oriented research

4.1.2. Type of tags

Tags in a tagset are classified into

- **Major Word Class Tags**: these tags comprise noun, pronoun, adjective, article, pre/postposition, numeral, conjunction, interjection, adverb and verb. These are the basic tags required in any tagset.

- **Punctuation Tags**: Punctuations like period (.), question mark (?), comma (,) etc are marked as distinctive tags .These tags mark the boundary of sentences, clauses and phrase. Punctuation marks helps in deciding the pattern of tag arrangement and they also help for the consistency in the annotation of the corpus.

- **Open or Unique Word class Tags**: In any language there is a set of unique grammatical elements like infinitive particles, negative particles etc., which do not come under any of the major word classes. These tags are significant and very helpful in the annotation for pattern matching.

- **Residual or Miscellaneous Tags:** In this category comes symbols other than punctuation marks, abbreviations, foreign words and other tokens which are not covered by the above three tag groups. In some tagsets, punctuation tags are also grouped together with miscellaneous tags.[166]

The above classification is for general guidelines to develop a tagset as to what all tags one should cover. But the design of the tagset heavily depends on the purpose and the methodology of the application in which it will be used. It also depends upon the nature of the language, like as in the case of Sanskrit there is no category of articles. Chinese language does not have a separate class for adjective, but that becomes a subclass of verbs. Even in Sanskrit the distinction between the noun and adjective is very subtle which will be discussed later. But the Sanskrit linguistic tradition does categorize adjectives as a separate class.

4.1.3. Major word classes

The major word classes like nouns, pronouns etc., can be further classified.

- Nouns can be sub-classified into Common Nouns and Proper Nouns

- Pronouns into Personal Pronouns, Demonstrative Pronouns, Relative Pronouns, Reflexive Pronouns, Indefinite Pronouns and Possessive Pronouns

- Adjectives into Positive Adjectives, Comparative Adjectives and Superlative Adjectives

[166] Cloeren, Jon, 1999, ‘Tagsets’ in ‘Syntactic Word-class Tagging’, Kluwer Academic Publishers, Dordrecht, The Netherlands, p.38-39

- Adverbs also in the similar way: Positive Adverbs, Comparative Adverbs and Superlative Adverbs
- Numerals into Cardinal and Ordinal Numerals;
- Verbs into Main and Auxiliary, Tense, Mood, Transitive, Intransitive etc.

Semantic classification is also possible for these word classes , like adverb can be classified into temporal and spatial adverb.

4.1.4. Features or Attributes

Feature or attribute tags contain the information like gender (male, female, neuter), number (singular, dual, and plural), case (nominative, accusative etc.), person as in the case of verbs (first, second, third) etc.

4.2. Issues in Tagset Development[167]

This section deals with various issues regarding Tagset development.

a) Fineness Vs Coarseness

The features which will be used during the annotation process is the first thing to decide. Which aspect of the word is needed to annotated is decided in this phase. Only lexical aspect can be the tag or plurality, gender etc. are distinctly tagged. This is a really important contribution on the performance of tagger. Lexical tagsets are easy to tag and more efficient but they fail to give information about basic morph syntactic of language.

b) Lexical Vs Syntactic Category

In tagset development, it is necessary to decide that whether the word would be tagged according to its lexical category or syntactic category. If the word is tagged according to its lexical category then a consistency in the tagging can be observed, because the lexical category of a word is always same. Word can have a different part-of-speech according to its context in a sentence. So this aspect should be considered before the tagset development.

[167] Chandra nitish, Sudhakar kumawat, Vinayak srivastava, *Various tagsets for Indian languages and Their performance in Part of speech tagging,* Department of computer science & engineering, IIIT(BHU), Varanasi

c) New Tags V/S Tags from a Standard Tagger

During tagset design process which tag should be used is an issue. Either a total new set of tags can be coined or any slandered tagset can be used as reference. The modification according to the language needs and tagger requirement is necessary if a standard tagset is used.

4.3. Issues in Selecting a Tagset

Every language has its own features. For selecting a tagset to annotate it a researcher has to consider some of the points.

a) Conciseness

First of all the conciseness is the basic requirement for any tagset. A short label is more suitable than a full or a long label. Like PRP is better than Preposition or prep. Most of the tagset use minimal word for tags.

b) Perspicuity

In a tagset the label should be easy to understand. It's easy to understand 'prep' than 'in'. This is a really important aspect, because without it ambiguity can arise.

a) Decomposability

The tagset should be possible to decompose in different parts. For an example V_VM_VF. In above example this label is easy to understand that there is verb, main verb, and finite verb. Decomposability in a tagset gives a place to different features to be encoded in a tag by separate sub-stings. Decomposable tags help in better corpus analysis (Leech 1997) by allowing to search with an underspecified search string.

4.4. Annotation Scheme for Indian languages

4.4.1. IIIT tagset

The vary first IIIT tagset for Indian Languages was based in Penn Treebank tagset. There are total of 26 lexical categories has benn listed in guidelines for annotating Indian languages (Bharat et al., 2006 as cited in Choudhary, N., 2011). This tagset was developed at IIIT Hyderabad through workshops on NLP.

4.4.2. TDIL program

Ministry of communication and information technology has started TDIL project for creation of technological solutions for Indian languages. The aim of this project was to develop information processing tools, that can facilitate human-machine interaction. Creating multilingual technology resource was also the goal of this project. Various institutes were the part of this project. The project was expanded to different research institute for several purposes e.g. English to Indian language MT system ran at CDAC Pune and IIT Kanpur, IIIT Hyderabad look after Indian to Indian Languages machine translation, HCU for Sanskrit-Hindi MT system, handwriting recognition by I.I.Sc. Bangalore, Cross-lingual information access by IIT Bombay and corpora development to JNU (Jha, 2010).

4.4.3. IL-POST

IL-POST (Indian Language – Part of Speech Tagset) is a hierarchical tagset, following EAGLES guidelines. EAGLES guideline is hierarchical in nature representing features and values of the grammatical categories. This tagset was developed by Microsoft Research India, Bangalore in 2008 in the course of developing annotated corpus for Indian Languages. Baskaran et.al. (2008).

4.5.Tagsets for Sanskrit

There are few annotation schemes developed by different institution for the tagging of Sanskrit text. This section of the chapter gives a description about available tagsets for Sanskrit. Following tagsets are developed or adopted according to linguistic feature of Sanskrit :-

4.5.1. ILMT Tagset (IIIT-Hyderabad)[168]

There is a Standard POS tagset designed at IIIT-Hyderabad named ILMT tagset. In this work a guideline for POS tagging and Chunking for Indian Languages has been developed. A collection of Application Program Interfaces (APIs) had been brought out. A tool for NLP called Sanchay has been developed based on this tagset, which also includes syntactic annotation interface. This Penn Tagset derived tagset is a flat Tagset. For making Penn tagset

[168] Gopal Madhav, Diwakar Mishra, and Devi Priyanka Singh "Evaluating Tagsets for Sanskrit", Lecture Notes in Computer Science, 2010

suitable for Indian languages certain changes have made. New tags were introduced or existing tags modified, where Penn tags were not adequate for Indian languages. For making Machine learning easier and give higher accuracy and coverage, less number of tags has been used. This tagset covers mainly lexical categories and does not consider syntactic function of the word in the sentence. The reason behind this is to reduce confusion coming during manual tagging. A word-tag relation has been established by the machine, which results in efficient machine learning. Only coarse analysis is allowed in this tagset. The fine-grained distinctions among linguistic categories have been avoided. "As too coarse an analysis is not of much use", keeping this fact in mind the designer of this tagset had tried to maintain a balance between fineness and coarseness. According to the designer of this tagset, the analysis of the data should not be as fine as it start hampering the machine learning. It should also not be so coarse that it loses important information. For that this tagset does not include gender, number, person etc as an attributes of lexical categories. Incorporated adjective and adverb related tags are not applied for Sanskrit. For making this tagset suitable for Sanskrit needs some efforts. For example; case of Particles. Like noun and verb, Particles needs more tags as they have valuable and various function in Sanskrit. Sanskrit, having a complex morphology makes the word sense disambiguation, difficult using this flat tagset.

4.5.2. JNU-Sanskrit Tagset (JPOS)[169]

POS tagging problem for Sanskrit is discussed for the first time in the doctoral thesis of R. Chandrashekar titled Part-of Speech Tagging for Sanskrit. In this work for preparing a rule based tagger for Sanskrit a tagset named JPOS has been developed using extensive Pāṇinīan grammatical description.

The classification of Sanskrit grammatical categories in JPOS is very exhaustive. The tags which are used in this tagst are very unique as the apply the Sanskrit grammatical terms. The main categories tag description is given. The tagset has tolal 134 tags. The tagset has 65 word class tags, 43 feature sub-tags. There is 25 punctuation and one tag for tagging unknown word. A combination of word class tag and feature sub-tag makes a single full tag. Indeclinable category and punctuation tags do not have any sub-tag.

The ambiguity resolution become easier with this as in Sanskrit, there is an agreement between the noun and verb and person and number.

[169] Ibid.

This is a flat and fine-grained tagset. It tries to capture linguistic features of a word according to the traditional Sanskrit grammar. This tagset has a high coverage. Every kind of linguistic entity was was treated discretely.*taddhita* and *kṛdanta* constructions were prominent in that.

This tagset is very informative because of its specificity. But same time this specificity crate problem suffer while tagging the data. Due to huge number of tags with extensive knowledge of *Pāṇinīian* grammar; the human annotator will have enormous cognitive pressure during tagging data.A well trained scholar of *Pāṇinīian* grammar can only handle this task.Thhe efficiency of the tagger has not been checked. Adoption of this tagset for tagging language other than Sanskrit is quite a difficult work as it will need an enormous change in the tagset. Because of that the tagged multilingual corpora will hardly be compatible with each other.

4.5.3. LDCIL Tagset[170]

Linguistic Data Consortium for Indian Languages (LDC-IL) is a consortium for the development of language technology in India. This consortium was set up by a an effort of the Central Institute of Indian Languages, Mysore and various other similar minded institutions, which are working in the field of Indian Languages technology. The LDCIL tagset is available in the website of Linguistic Data Consortium for Indian Languages: *http://www.ldcil.org/POSTagSet.html.* The standardization committee formulated the tagset on August 6, 2007 at IIIT, Hyderabad. This tagset is an advanced extension of ILMT tagset, with few additional tags, to increase the granularity. This tagset is a standard tagset for tagging Indian language corpora. It is a flat tagset and has 26 tags.

4.5.4. IL-POSTS Sanskrit Tagset[171]

This tagset has been developed after changing some category in IL-POSTS, which was a standard framework for Indian languages and developed by MSRI. This is a hierarchical tagset. This tagset is based on similar guidelines to EAGLES.

Categories, Types, and Attributes are the three levels where the information is encoded.except Sanskrit three more languages has adopted tagsets from this framework, these

[170] Ibid.
[171] Ibid.

are Hindi, Bangla and Tamil. Linguistic richness of Indian languages has been taken care of by this tagset. For the purpose of making the framework a common standard across languages and project, the selective inclusion and removal of features for a specific language/project has been done. Cross-linguistic compatibility, reusability, and inter changeability among languages is the goal of this tagset.This tagsets has achieved a higher accuracy result. This framework covers the linguistic features of Indian languages, makes it very sophisticated.

The extreme fine-grained tags of this framework, encodes extensive information about the gender, number case and other linguistic feature. All the desired linguistic features of Sanskrit are properly accommodated in IL-POST. The tagged corpus in this framework remains compatible with other language. This framework follows the European language standard, which enables user to access the data in European language for Indian NLP and vice versa.

4.5.5. Sanskrit Consortium Tagset (CPOS)[172]

For annotating sandhi-free Sanskrit corpus, this tagset has been designed. It follows traditional Sanskrit grammatical categorization. The extensions of yāska's four grammatical categories: *nāma* (noun), *ākhyāta* (verb), *upasarga* (preverb), and *nipāta* (particle) are the basis of this tagset. This flat tagset has 28 tags. It annotates only major categories. JPOS or ILMT tagset's categories have been taken. Morpho-syntactic features are not included in the tagset because the tagsets take help of a Morphological Analyzer for analysis. As it has less number of categories, it is assumed that the tagset give higher accuracy. To help manual tagging a tool named 'Sanchay' has been used. If the purpose of the tagged data is known then it is useful for Sasnkrit tagging. Using this tagset, two taggers; a rule based and a statistical were developed which achieved an accuracy of 79-81%. The sharing of data and its reusebality is not possible as the framework does not follow any standard.

At an experimental level two taggers were developed using this tagset - a rule based and a statistical and have reported 79-81% results. The rule based tagger can be accessed using this link http://sanskrit.jnu.ac.in/cpost/post.jsp.

[172] Ibid.

4.5.6. The BIS Tagset by DeitY for ILCI[173]

The Bureau of Indian Standards (BIS) Tagset has recommended the use of a common tagset for the part of speech annotation of ILs. But there was no consensus on the POS tagging schema for the Indian Languages till very late. Though some POS taggers were developed for languages like Shrivastava, M. & Bhattacharya, P., for Hindi (2008), Avinesh, PVS & G. Karthick for Telugu (2007), Dandpat, S. et.al. for Bengali (2007) etc (as mentioned in Choudhary N. and Jha G.N., 2011). ILMT (Indian Language Macune Translation) also claims to have developed different taggers but they was not a big success. BIS (Bureau of Indian Standards) and ILCI made the first national standard POS annotation. This tagset is a result of the POS Standardization Committee appointed by the DeitY, Govt. of India. This hierarchical tagset coarsely allows only two levels of POS categories (sometimes to three levels, at max), the major categories and sub-categories. MSRI and ILMT categories are included to a great extent, in the tagset. It is a comprehensive tagset with no morpho-syntactic features. It is capable of sharing, reusing and interchanging the linguistic resources and captures appropriate linguistic information (Gopal, 2012).

This tagset is a national standard tagset for Indian languages. BIS has been recently designed by the POS standards committee at IIIT, Hyderabad. It was finalized on June 12, 2010. There is 11 categories in the top level of this tagset. Further the top level category have subtype level 1 and subtype level 2. This tagset follow a standard which can handle the linguistic richness of Indian languages. This is a hierarchical tagset. This tagset annotate the major categories. BIS tagset is also take care the types and subtype of the major categories. The granularity of linguistic information is directly related to the hierarchy of tags. As much deep the hierarchy of the tags is in the tagset, more fine grained linguistic information can be gained. In BIS the granularity of the POS is coarse. Because of that, most POS categories have the hierarchy of two levels. In this tagset the highest level of hierarchy of the POS is three levels. Presently, this tagset is being used for parts of speech annotation by the ILCI Corpora Project under the TDIL programme.

[173] Ibid

4.6. Sanskrit Annotation Using the BIS Tagset

Sanskrit is world's one of the classical, well-studied language. It has a very sophisticated vocabulary, rich morphology, wide literature, millennia old scholarship and most importantly a great rich grammatical tradition. There is eminence literature is available on morphology and phonology of Sanskrit which describing it variously. However, syntax of Sanskrit has not gained much attention of linguists. The morphological categories based discussion of Sanskrit has a long tradition but syntactic categories have rarely draw attention of the linguists.

As stated by Gopal and Jha et.al. (2011); "POS tagging, however, is a mix of morphological and syntactic description of a language and rightly called morpho-syntactic tagging also.The morphological categories of Pāṇinian grammar are, however, very wide and less formal from syntactic point of view. Therefore, they need to be further categorized and described in terms of contemporary morpho-syntactic categories of linguistic description."

Because of this necessity the BIS tagset guideline for Sanskrit is prepared by Madhav Gopal and Dr. Girish Nath Jha. This guideline is given in detail here :-

4.6.1. List of List of Annotation Labels with Corresponding Parts of Speech

The present section contains the annotation labels and their corresponding parts of speech alphabetically.

Annotation labels	**Parts of speech**
1. CC_CCD	Coordinating conjunction
2.CC_CCS_UT	Quotative
3.. CC_CCS	Subordinating conjunction
4.. DM_DMD	Deictic demonstrative
5. DM_DMI	Indefinite demonstrative
6.. DM_DMQ	Interrogative/wh demonstrative
7. DM_DMR	Relative demonstrative
8. JJ	Adjective
9. N_NN	Common noun
10. N_NNP	Proper noun
12. N_NST	Spatio-temporal noun
12. N_NNV	Verbal noun
13. PSP	Postposition
14. PR_PRC	Reciprocal pronoun
15. PR_PRF	Reflexive pronoun
16. PR_PRI	Indefinite pronoun
17. PR_PRL	Relative pronoun
18. PR_PRP	Personal pronoun

19. PR_PRQ	Interrogative/wh pronoun
20. QT_QTC	Cardinal quantifier
21. QT_QTF	General quantifier
22. QT_QTO	Ordinal quantifier
23. RB	Adverb
24. RD_ECH	Reduplicative echo word
25. RD_PUNC	Punctuation
26. RD_RDF	Foreign word
27. RD_UNK	Unknown word
28. RD_SYM	Symbol/special character
29. RP_CL	Classifier
30. RP_NEG	Negation
31. RP_INJ	Interjection
32. RP_INTF	Intensifier
33. RP_RPD	Default particle
34. V_VAUX	Auxiliary verb
35. V_VM	Main verb
36. V_VM_VF	Finite verb
37. V_VM_VINF	Infinitive verb
38. V_VM_VNF	Non-finite verb
39. V_VM_VNG	Gerundive verb

Table 4.1. Annotation label with their POS

4.6.2. Parts of Speech Description with Their Corresponding Annotation Labels

This section contains the description of the parts of speech in a detailed manner along with examples from the data used in developing the Statistical Parts of Speech Tagger for Sanskrit. They are as follows:-

1. **Noun (N)**

By the standard definition, Noun is the names given to a person, place or thing. The nouns have three sub-divisions namely common noun, proper noun and temporal nouns.

In the BIS annotation scheme noun is the top level category which is further divided in four subtypes at level 1: common, proper, verbal and noun location.

1.1 Common noun (NN)

Different kinds of nouns have been annotated as common nouns such as common, abstract, material, countable and non-countable nouns etc. For example any person, place or a thing, emotions, ideas etc, group of things, animals, or persons.e.g.

Sarveḥ/PRP ***janāḥ*/NN *āpaṇaṃ*/NN** gacchanti/VF |/PUNC

1.2 Proper noun (NNP)

Where a word denotes about a specific name of a person, institution, , shop, date, day, month, place, species, etc., name of anything is considered as a proper noun. Whenever a word is used as a proper noun in any context then it will be tagged as proper noun, whatever category it belongs.

If the word is of some other category, but is used as a proper noun in a context; should be marked as proper noun. e.g.

suṣamā /NNP gorakhapuram/NNP *gacchati*/VF |/PUNC

1.3 Verbal noun (NNV)

Tamil and Malayalam language have verbal noun. This feature is also applicable for Sanskrit. In this category *kṛdantas* like *āgamanam* and *hasanam* could be annotated as verbal noun, when they are nominalized in any sentence.e.g.

śrī /NNP obāmāḥ /NNP ***āgamanam* /NNV** *kadā* /PRP *bhaviṣyati*/VF ?/PUNC

tasya /PRP ***hasanam* /NNV *śobhanam* /NNV** *asti*/VF ।/PUNC

1.4 Nloc (space and time) (NST)

In noun category the fourth sub-type is related with space and time, is Nloc. The annotation label for this category is NST. Distinctive nature of some locational nouns is registered by this category. These locational nouns also functions as a part of complex postpositions. Indeclinables like *agré* and *pūrvam* could be annotated as noun location.e.g.

***agre* /NST** *vānarāḥ* /NN *miliṣyanti* /VF ।/PUNC

*Teṣām/*PRP ***pūrvam* /NST** *duṣṭāḥ* /NN *api* /RPD *militum* /VINF *śaknuvanti* /VF ।/PUNC

2. Pronoun (PR)

Pronouns are those words which are used in the place of a noun, that has been already got mentioned. Often the purpose of using pronoun is to avoid the repetition of noun.

IN BIS tagset the pronoun category has 5 subtypes. These are personal pronoun, reflexive pronoun, relative pronoun, reciprocal pronoun, and wh-word.

2.1 Personal pronouns (PRP)

Personal pronouns are those word which are used in the place of personal noun. In other words those nouns which are referring to a specific person or things are replaced with personal pronoun to avoid the repetitions.

e.g., personal pronouns: ***aham, tvam, bhavān, saḥ*** etc.

Inclusive Pronouns: ***sarvam, ubhayam*** etc.

Indefinite pronouns***: kaścit, kiṃsvit,*** etc.

For example

***saḥ*/PRP** *syūtam* /NN *ādāya* /VNG *pāṭhaśālām*/NN *gacchat* /VF ।/PUNC

2.2 Reflexive pronouns (PRF)

A reflexive pronoun comes after the noun or pronoun whichever it refers. These noun and pronouns are its antecedent. In Sanskrit; words like *ātman, svayam* etc are used as reflexive pronoun.e.g.

rājā svayaṃ samarabhūmiṃ jagāma "the king himself went to the battlefield".

The nominal words like *ātman, sva, svakīya* etc. could be annotated as reflexives.e.g.

bālikā /NN vidyālayam /NN **svayam /PRF** gacchati /VF ।/PUNC

2.3 Relative pronoun (PRL)

"The relative pronouns are those pronouns whose antecedent can be either a noun or a pronoun. However, these pronouns do not make any difference in number or gender as in the case of personal pronouns (ILCI, 2010). For example; *yat* (which), *yam* (whom), and *yā* (who-fem).e.g.

***Yasya* /PRL** *na* /NEG *asti* /VF *svayam* /PRF *prajñā* /NN *śāstram*/NN *tasya* /PRP *karoti*/VF *kim/*PRQ ?/PUNC

2.4 Reciprocal pronoun (PRC)

Repetition of the pronominal adjectives is the expression of reciprocity e.g. *anyonya, itaretara, ekaika*, and *paraspara*. These words are commonly used in the singular form. Detarmination of their gender is done by considering the gender of their referents. Reciprocal pronoun's morphological number and nominal declension and their case should be marked if it is relevant according to context, else their marking should not be applicable. The major reciprocal pronouns are *parasparam*, *itarétaram*, *mithaḥ* and *anyonyam*.

bālakāḥ /NN ***parasparam* /PRC** *krīḍanti* /VF ।/PUNC

2.5. Wh word (PRQ)

The interrogative pronouns are the pronouns that are used to ask questions. Wh- Pronouns like *kaḥ, kim* etc. comes in this category.e.g.

Nāyakaḥ/N_NN ***kim*/PRQ** *karoti*/V_VM_VF | RD_PUNC

3. Demonstrative (DM)

In BIS, the third top level category is demonstrative. Demonstratives, are those pronouns which necessarily followed by a noun, another pronoun or adjective. This category has three sub-type; deictic, relative and wh-word.

3.1 Deictic demonstrative (DMD)

To differentiate pronouns (PR) from demonstratives (DM), they have been tagged as pronouns when their referents exist beyond the same phrase boundary. In other words, if the referent is in the preceding phrase, then the pronoun can come in the other following phrase. Demonstrative pronouns and third person pronouns are same in Sanskrit. e.g.

***tat* /DMD** *gṛham* /NN sumedhasya /NNP *asti* /VF ,/PUNC *idam* /DMD *gṛham* /NN Lalitasya /NNP ।/PUNC

3.2 Relative demonstrative (DMR)

According to ILCI, 2010; “The relative demonstrative occurs in the same form as the relative pronoun. The difference is only that these relatives are always followed by a noun that it modifies”. e.g.

***yā* /DMR** *bālā* /NN *tatra* /RB *krīḍati* /VF *sā* /PRP *nṛtyāṅganā* /NN *api* /RPD *asti* /VF ।/PUNC

3.3 Wh-word demonstrative (DMQ)

The wh-demonstratives are the same wh-words (or question words) which act as wh pronouns. The difference is that in their demonstrative function, they do not ask a question, rather only demonstrate.Wh demonstrative is followed by a noun, pronoun or adjective. e.g.

***kaḥ* /DMQ** *bhikṣukaḥ* /NN *bhikṣām*/NN *icchati*/VF ?/PUNC

4. Verb (V)

In BIS framework the verb category is complicated. Firstly it divides in main verb and and auxiliary under subtype level 1. Then under subtype level 2 it has finite, non-finite, infinitive and gerund divisions. According to Gopal and Jha (2011); "Verb main does not seem to be an appropriate tag in case of Sanskrit. However, if anybody insists to use it, it can be utilized in tagging the verbs of present tense when followed by a *sma* and the *kta* and *ktavat pratyayāntas* when followed by an auxiliary, and in doing so the auxiliary verbs and *sma* have to retain their Auxiliary tags."

4.1 Main verb (VM)

As told above, in Sanskrit this tag is not applicable but can be used if the scholars think to apply this tag. Verbs of present tense followed by a *sma* can be tagged with this label. The *kta* and *ktavat pratyayaantas* followed by an auxiliary can be also tagged with this tag. When this tag is used, the auxiliaries have to retain their VAUX tag.

4.1.1 Finite (VF)

In finite verbs (VF), all the conjugations of *dhātus* are included. "However, when some of these forms will be used to express the aspectual meaning of the preceding *kṛdanta* will be tagged as auxiliary, as is stated above. In addition, *kta* and *ktavat pratyayāntas* will also be tagged as VF when they are not followed by an auxiliary.

As we do not have a separate tag for gerundives (like *kāryam*, *karaṇīyam*, *kartavyam*), VF tag could be applied for them as well" (According to Gopal& Jha (2011); e.g.

mohanaḥ /NNP Haidarābādam /NNP ***gatavān* /VF** ।/PUNC *saḥ* /PRP *mama* /PRP *bhrātā* /NN *asti* /VF ।/PUNC suṣamā /NNP Viśākhāpattanam/NNP ***gacchati* /VF** ।/PUNC

4.1.2 Non-finite (VNF)

According to Gopal and Jha (2011); "*kta* and *ktavat pratyayāntas* will be tagged as verb non-finite (VNF) when followed by an auxiliary and other *kṛidantas* like *śatṛ*, *śānac* and *kānac* will also get the same tag". e.g.

suṣamā /NNP Prayāgam /NNP ***gacchantī* /VNF** *tena*/PRP saha /PSP *vārtām* /NN *kariṣyati* /VF ।//PUNC

Adhunā /RB sā /PRP Siṃgāpuram /NNP *gatā* /VNF *asti* /VAUX ।/PUNC

4.1.3 Infinite (VINF)

The infinitives of Sanskrit are different from English other Indian languages. in English they correspond to the infinitive of purpose. But in Sanskrit only *tumun pratyayāntas* will be tagged as VINF. e.g.

sā /PRP Jayapuram /NNP api /RPD **gantum /VINF** icchati /VF ।/PUNC

4.1.4 Gerund (VNG)

In the Sanskrit literature gerund are *ktvānta* and *lyabanta*. Therefore, these words will be annotated with verb non-finite gerund (VNG). e.g.

Suṣamā /NNP *gorakhapuram* /NNP ***gatvā* /VNG** Prayāgam /NNP *gamiṣyati* /VF ।/PUNC

Tatra /RB *ca* /CCD svakīyām /PRF mātaram /NN **ādāya /VNG** gaṅgāsnānam /NN kariṣyati /VF ।/PUNC

4.2 Auxiliary (VAUX)

According to Gopal and Jha (2011); "The *tiṅantas* (inflections of as, *ās, sthā, kṛ, bhū* only) that follow a *kṛdanta* to express its (*kṛdanta's*) aspectual meaning, will be tagged with

Auxiliary label and the indeclinable *sma* will also get the same tag when follows a verb in present tense and modifies the meaning of the associated verb”. e.g.

Tataḥ /NST ca /CCD Piṅgalakaḥ /NNP *sañjīvakena* /NNP *saha* /PSP *subhāṣitagoṣṭhīsukham* /NN *anubhavan* /VNF *āste* **/VAUX ।**/PUNC

Tasmin/DMD *vane* /NN Bhāsurakaḥ /NNP *nāma* /JJ *siṃhaḥ* /NN *prativasati*/VF **sma/VAUX** ।/PUNC

Saḥ /PRP *adhunā* /RB Siṃgāpuram /NNP *gataḥ* /VNF ***asti*** **/VAUX ।**/PUNC

5. Adjective (JJ)

An adjective is that word which modifies a noun. Adjectives are generally followed by nouns. There is two types of adjective. Adjective are two types; attributive adjective and a predicative adjective. In Sanskrit, adjectives are seldom realized as modifiers. Most of the time, they occur as substantives. The depth of pure adjective uses is not easy to decide. They are annotated as adjectives when they are used with their modified items, else they should be tagged as noun. e.g.

dhīrodāttaḥ **/JJ** *nāyakaḥ* /NN *kalaham* /NN ,/PUNC *īrṣyām* /NN *ca* /CCD *na* /NEG *karoti* /VF ।/PUNC

6. Adverb (RB)

In BIS framework, only manner adverbs are to be annotated. Because of it *uccaiḥ* (loudly), *sukhaṃ* (happily) etc. will be tagged with the adverb tag. e.g.

yānam /NN ***vegena*** **/RB** *gacchati* /VF ।/PUNC

7. Postposition (PSP)

In BIS Postpositions is a top level category. According to Gopal and Jha (2011); “Sanskrit does not have postposition as such. But we can tag the *upapada* indeclinables as postpositions as they are indeed ambipositions and cause the assignment of a particular *vibhakti* in the concerned nominal”. e.g.

Durgam /NN ***abhitaḥ*** **/PSP** *parikhā* /NN *asti* /VF ।/PUNC

8. Conjunction (CC)

The BIS tagset consider conjunction as one of the major category. It has co-ordinator, subordinator and quotative as subtypes.

8.1. Co-ordinator (CCD)

Those conjunctions which join two or more items of similar syntactic importance are known as Co-ordinators. These will be assigned with CCD label. In this list words e.g. *ca, api ca, tathā ca, tathā* are included.e.g.

nāyakaḥ /NN *khalanāyakaḥ* /NN ***ca*** **/CCD** *saharūpeṇa*/UNK *gacchanti* /VF ।/PUNC

8.2. Subordinator (CCS)

According to Gopal (2011) "The conjunctions that introduce a dependent clause are subordinators. The conjunctions *yat*, *yena*, *yadi* etc. will be labelled as CCS".e.g.

rāmaḥ /NNP *akathayat* /VF ***yat*****/CCS** *saḥ* /PRP *āpaṇam* /NN *gamiṣyati* /VF ।/PUNC

8.2.1. Quotative (UT)

'Quotatives' are the sub-type of subordinators. They occurs in many languages. Their role in those languages is to conjoining a subordinate clause with the main clause. This is the reason that it is the third level of hierarchy in conjuncts. Although it is optional to any language to go to this level of granualarity. e.g.

"/PUNC *sarve* /PRP *bhavantu*/VF *sukhinaḥ* /NN "/PUNC ***iti*** **/UT** *kena* /PRQ *uktam* /VF ?/PUNC

9. Particle (RP)

According to Gopal & Jha (2011); "Particle is a very important category for Sanskrit language, as they have many a role to play. Some of the indeclinable described as *avyayas* in the grammar fall in this category. In the tagset, there are default, classifier, interjection, intensifier and negation subtypes of the Particle category. The classifier tag is not applicable for Sanskrit".

9.1 Default (RPD)

In the BIS system for Sanskrit, this would be applied for all avyayas, which lacks specific tag in this framework. *avyaya, sādṛśyādi, avadhāraṇam*, and *praśnārthaka* are included in this.

Example 1

***Atha* /RPD** *kim* /PRQ *karaṇīyam* /VF ?/PUNC *sukumārā* /JJ ***khalu* /RPD** *iyam* /PRP ?/PUNC

Example 2

api /RPD *gacchati* /VF *saḥ* /PRP ?/PUNC

Example 3

***ām*/INJ** ,/PUNC *saḥ* /PRP *eva* /RPD *gantum* /VINF *śaknoti* /VF ।/PUNC

9.2 Interjection (INJ)

Words which express emotions are termed as interjections. In this category all the particles which one use for getting the attention of people, e.g., *bata, aho, hā, dhik, svadhā, he, bho* etc are include. e.g.

***bho*/INJ** *narāḥ* /NN !/PUNC *yūyam* /PRP *kim* /PRQ *kurutha* /VF ?/PUNC

9.3 Intensifier (INTF)

Intensifiers are the adverbial elements with an intensifying role. They could be both, either positive or negative. *bhṛśam, pūrṇatayā, nyūnatayā, nyūnātinyūnam* etc. will fall in this category.e.g.

tvaṃ/PRP *kathaṃ/*PRQ ***bhṛśam* /INTF** *cintayasi*/VF ।/PUNC

9.4 Negation (NEG)

Negation words come in category of indeclinable, but when these words express a negative meaning, and then they are tagged with this tag. e.g.

Cintā /NN ***mā*** **/NEG** *karotu* /VF ।/PUNC *saḥ* /PRP *bhavantam* /PRP *na* /NEG *tāḍayiṣyati* /VF ।/PUNC

10. Quantifier (QT)

Those words which quantify the noun are called quantifiers. It gives information that the noun is in definite number or indefinite number or denotes the amount of that noun e.g. *Daśam, Tṛtīyaḥ, Katipaya, Sarve*. In this category; there are three subtype; general quantifiers, cardinal quantifiers and ordinal quantifiers. These tags are equally applied to different type of quantifier; written in words (like four, fourth etc.) and written in digits (like 4, 4th etc.)

10.1 General quantifier (QTF)

If a word does not indicate a definite amount or quantity then it comes in general quantifiers. Words like '*Sarv', 'katipaya'* etc are general quantifier. e.g.

Sarveḥ/QTF *bālakaḥ*/NN *pushtakaṃ*/NN *pathanti*/VF ।/PUNC

10.2 Cardinal quantifier (QTC)

In this category comes those numbers which quantify and gives a precise amount of objects. These are known as cardinal quantifiers. e.g.

***Daśa*/QTC** *grāmasya*/NN *janasaṃkhyā*/NN *nirvācane*/NN *sammilatāḥ*/VF।/PUNC

10.3 Ordinal quantifier (QTO)

Ordinal quantifiers are those words, which gives information about the order of a particular object that where it is placed. e.g.

asya /PRP *lekhakasya* /NN *prathama* **/QTO** *pustakaṃ* /NN *atiprasiddham* /JJ *asti* /VF
।/PUNC

11. Residual (RD)

In BIS residual is one of the major categories. It has further categorized in five subtype; foreign word, symbol, punctuation, unknown and echo words.

11.1. Foreign word (RDF)

Any word which is written in any other script other than Devanagari then it is considered as a foreign word, and will be tagged with this tag. e.g.

pañca/QTC *medhāvinaḥ*/JJ *bhāṣāvijñāḥ*/NN **POS/RDF *tagging*/RDF** *iti*/UT *kurvanti*/VF
।/PUNC

11.2. Symbol (SYM)

The symbols are the mathematical or other special characters that are not part of the

regular Sanskrit language. This subtype of residual is for symbols like $, %, # etc. e.g.

Mama/PRP *pāragamanapatrasya*/NN *mūlyam*/NN **$/SYM** 500/QTC *asti*/VF ।/PUNC

11.3. Punctuation (PUNC)

Punctuations include the characters that are considered as the regular punctuation marks in Sanskrit. All the punctuations marks will be annotated with this tag, therefore all other symbols can be tagged as symbol. e.g.

Sureśaḥ/NNP **,/PUNC** Raviḥ /NNP **,/PUNC** Vaibhavaḥ /NNP *ca* /CCD *nagare*/NN *gacchanti* /VF **।/PUNC**

11.4. Unknown

In this category all those words come that cannot be decided by the annotator. These may include words and phrases or sentences from a foreign language written in Sanskrit. e.g.

Nāyakaḥ/NN *khalanāyakaḥ*/NN *ca*/CCD **Saharūpeṇa /UNK** vasataḥ /VF ।/PUNC

11.5. Echo Words (ECH)

According to Gopal & Jha (2011)" The name of this category should be Reduplication as this is a broader one. In Sanskrit echo words are mainly reduplications of a variety of linguistic items". In this category the first part will be tagged as RB and the reduplicated form will get ECH. e.g.

śanaiḥ /RB śanaiḥ /ECH agre/NST *calāmaḥ* /VF ।/PUN

So as we see how BIS tagset can be used for tagging Sanskrit language.

4.6.3. Advantage of using BIS Tagset

Tagging with a standard tagset makes corpora suitable for uniform use. It will remove the language barriers between different languages communities. This will also maximize the chance of use and sharing of tagged data. Learning the differences and similarities in different languages, linguistic and natural language processing researchers will be benefitted in their studies. The corpus which is annotated with this tagset would be more beneficial as it is annotated by a standard tagset or paradigm. Indian languages can enhance their linguistic resource by using these standards for their corpora annotation program.

4.6.4. Issue & Improvement in BIS Tagset[174]

BIS tagset by any means is not perfect. It is still not comprehensive and unambiguous so its needs constant fine-tune. BIS tagset can be improved in the following way

1. Expand the definition of Adverb tagset
2. Incorporate tagsets to mark all the words of a proverb or idiom together
3. If more than one main verb is found in a sentence let it be distinguished as VM1 and VM2
4. List of words under NLOC must be increased.

[174] Chandra nitish, sudhakar kumawat, vinayak srivastava, *various tagsets for indian languages and their performance in part of speech tagging*, department of computer science & engineering, iit (bhu), varanasi

4.7. Preparation of Data recourse-

This part of the chapter has two sections. The first section deals with Data collection method and the second part discuss the process of linguistic resource preparation. The linguistic resources have been developed by three major processes for modeling of the systems. They are annotation, validation, and tokenization.

4.7.1. Challenges in Annotation of Sanskrit

The Sanskrit language offers a quite interesting challenge for computational linguists. The Sanskrit language was studied to a high degree of formalization from high antiquity by genius linguists such as Pāṇinī. There is a continuous tradition of commenting and refining the primer work still very much alive. Because of that there is a great hope for the emergence of new computational models of linguistic treatment that can be well tuned by definition to the Sanskrit language.

The Sanskrit corpus is enormously huge. It contains a wealth of knowledge and wisdom from every sphere of human intellect. This wisdom of Sanskrit intelligence has not been yet properly brought to light, despite centuries of both Western and Indian scholarship. There has been a lot of work done in the field of NLP considering Indian language and the successes rate is also satisfactory. But Sanskrit has still not fully processable due to its complex grammatical sentence formation. The computational challenge of even simple tasks such as part-of-speech tagging has been a proven a stumbling block, which is hampering the use of computers in the processing of Sanskrit text for even simple philological tasks, for example finding all forms of a given root or morphological derivation in text is quite challenging.

- While tagging a Sanskrit corpus, an annotator faces many unique challenges which is specific to Sanskrit only, and not seen in other language data. These challenges are :
- First of all, identification of exact category for any word is the main problem with the annotator trained in traditional grammatical system. Most of the time the use a generic tag while a specific tag should be used to tag that word.
- The case of samasa and sandhi is also very crucial. Generally Sanskrit language tokens are written with no explicit boundaries. This happens because of the *sandhi*

process (euphonic combination of the words). So before annotation, the text needs to be analyzed by a *sandhi* analyzer, which is based on Pāṇinī formalism.

- Sanskrit has a unique feature that makes it hard to identify the grammatical category or boundary of sentence. This feature is 'the absence of punctuation marks'. Although , now a days the modern edition of the Classical works, modern Sanskrit literature have started using punctuation marks. The electronic version of Sanskrit text is using punctuation marks properly. As spaces and punctuation are used for token delimiters where 'multi-word' tokens and 'contracted form' tokens (in this category abbreviations, acronyms, etc. are included).
- Sanskrit, being a classical language has a lot of poetic expression. Even prose and scientific text are full with poetic expression. These expressions are hard to decode.
- Sanskrit is a free word order language, which implies that no fixed order is imposed on the word sequence. This creates difficulties for a statistical tagger as many permutations of the same string are possible. Additionally, by combining various morphemes, several words can be generated, which may not be present in the reference resources.

4.7.2. Corpora Annotation

During the period of training approximately 155k number of tokens has been taken in the experiment. In both the seen and unseen data from the phase I, data of literature domain has been taken. In literature domain the tokens has been taken from the text *Panchatantra,* 77k tokens. On the other hand, in phase II, entertainment, news and literature comprise a total of 78k tokens data.

Phase	**Source**	**Tokens**
First phase	*Panchatantra,* Various blogs	77,000
Second Phase	Sudharmā Newspaper and other sites.	78,000

Table 4.2 Data collected for Training

It is obvious that during annotation of the data, the annotators commit errors and gradually, they increase their efficiency by getting into contact with a large number of other unique constructions. It is indispensable that the annotated data has to be processed at least once to maintain both the linguistic quality and for the cause of efficiency of the statistical taggers. The reason for error-prone nature of the data during the annotation phase is, because of more than one human annotators involved in the process. They cannot agree in several of the instances of judgment as to how should be a 99 particular word tagged in a given context. These errors were taken care in the next step of validation process.

For further development of tagged corpora the Bootstrap method was used. According to Church (1988); "In this method at first, a relatively small amount of text is manually tagged and used to train a partially accurate model. The model is then used to tag more text, and the tags are manually corrected and then used to retrain the model. Church uses the tagged Brown corpus for training. These models involve probabilities for each word in the lexicon, so large tagged corpora are required for reliable estimation"[175].

4.7. 3.Validation of the Data

Validation process is required as in this process, all the annotated data have been validated and checked with care to ensure that there are no further errors or any undesired elements. The judgment to be taken at this stage is based on the context of the word. Thus context of the word has given priority for judging. If the tag of the word is not correctly labeled by the annotator then this stage that error get corrected.

4.7. 4.Tokenization

At this stage, the quality of the data is checked. For that the annotator checks the data and ensures that the data to be used for training is error free. Every item in the data should be separated with a tab. The automatic tokenizer tokenizes the data wrongly if there are any unnecessary spaces or no any space between the two tokens. Therefore, it is necessary to be quite cautious dealing with the punctuations in the data.

For example

[175] Church K. W, (1988) A stochastic parts program and noun phrase parser for unrestricted text. In Proceedings of the Second Conference on Applied Natural Language Processing (ACL), pages 136-143

Input token-

एकस्मिन् नागदन्ते लम्बमानः आसीत् |

Output token-

एकस्मिन्

नागदन्ते

लम्बमानः

आसीत्

After this process the data is ready. A statistical system learns from the language corpora, given to it as training set. Therefore, preparation of efficient corpora is the first and foremost necessity for any stochastic system.

CHAPTER 5:

TRAINING POS TAGGER, TESTING AND EVALUATION

Introduction:-

This chapter is the most important chapter of this work, as it deals with tagger training, testing and evaluation. The first part of the chapter gives an introduction to the theoretical part of SVM and its properties. The second part deals with the experiment set-up of the tagger. Third and fourth parts are related with Tagger's training and testing. User interface and technology used in development of the tagger is discussed in the fifth part of the chapter. Next part is the evaluation part, in which a descriptive analysis of the tagger's performance has been given. Suggestions for improvement in the tagger are given in the last part.

5.1. Support Vector Machines

Support Vector machine is a supervised learning model. It is generally defined as optimal margin classifiers. It was developed by Vapnik in 1995 at COLT92 (Computational Learning Theory) conference. SVM is based on the statistical learning theory and well known for its generalization performance. SVM is used for analyzing data, classification and pattern recognizing in indefinite feature space. Regression, here, means the extension of the machine to function approximation and the comparison of its performance with other function approximators (Kecman 2006).

In simple words, a Support Vector separates the two classes of variables by drawing hyperplane(s) in the features space. The data in each class are denoted with dots. The best hyperplane is the one which is at the farthest distance from the data point of any class, therefore, also called optimal margin classifier. Wider margin lowers the generalization error of the machine.

The reason for including this as the learning model for POS tagger is that it is used for classification and categorization of the input data in one of the two categories by training algorithm and making it a non-probabilistic linear classifier. Its extension to non-linear classifiers mapped on high dimension feature space is to fulfill the demand arising out of availability of very large number of electronic data under process every minute. This mapping on infinite space is possible with the use of Kernel function [$k\ (x,y)$].

A kernel function is used for calculating the dot product of two non linear vectors, mapped in the infinite feature space. Different kernel based methods derived in due course of time are namely Kernel Least Square Method, Kernel Principle Component Analysis and Kernel Mahalanobis Distance.

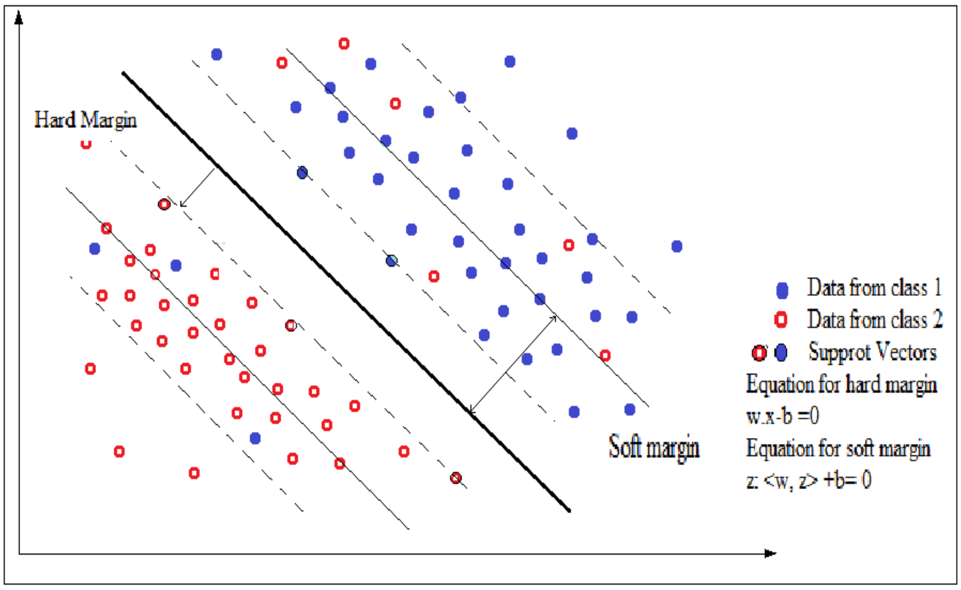


Fig. 5.1. SVM Classifier (classifying negative and positive examples)

Adapted from Gimenenez and Marquez (2006)

In the above figure 1, there are two sets of data and H1, H2 and H3 are three hyperplanes. The H1 hyperplane does not separate these classes, H2 does but the margin of separation are very less at different point and H3 separates the two classes with maximum margin for both linear and non-linear variables.

The set of *n* points can be calculated in a given training data *D*, as

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{R}^p,\ y_i \in \{-1, 1\}\}_{i=1}^{n}$$ Equation 1

Where, X_i and Y_i belong to same class and Y_i is either 1 or −1. Each X_i is a *p*-dimensional real vector. The maximum-margin hyperplane will divide the points having

Y_i= 1from Y_i= -1. **Equation 1**

Therefore, set of points satisfying X can be written as the following hyperplane:

$$\mathbf{w} . \mathbf{x} - \mathbf{b} = 0$$ **Equation 2**

Where, w is the normal vector to the hyperplane.

From the above equation we get the imaginary hyperplanes *m* and *n* as

$w \cdot x - b = 1$ and Equation 3

$w \cdot x - b = -1$ Equation 4

From geometry, the distance between the two hyperplanes comes out to be $2/||w||$. By adding the constraint, we can prevent the data point falling into the margin as we get

$y_i(w \cdot x_i - b) \geq 1$ Equation 5

Support Vector has been applied to various problems in many fashions. In the due course, the SVM is found to come up with the following variants:

(i) Least square SVM

(ii) Linear programming SVM

(iii) Robust SVM,

(iv) Bayesian SVM

(v) Committee machines.

Based on this basic principle of optimal margin, different problems have come up with various solutions. The equations derived for resolving the issues for different problems, with the help of SVM, are enlisted in the table 1 below:

a)	Quadratic programming optimization problem:
	$\arg\min_{(\mathbf{w},b)} \frac{1}{2}\|\mathbf{w}\|^2$ (a)
b)	Lagrange multipliers
	$\arg\min_{\mathbf{w},b} \max_{\boldsymbol{\alpha}\geq 0} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i [y_i(\mathbf{w}\cdot\mathbf{x_i} - b) - 1] \right\}$ (b)
c)	Karush-kunh-Tucker condition
	$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i}.$ (c)
d)	Dual form

	$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i}.$ (d)
	Where kernel is defined as $k(x_i\, x_j) = x_i \cdot x_j$
e)	Soft margin principle for mislabeled examples by Corinna Cortes is resolved with
	$\arg\min_{\mathbf{w},\xi,b} \max_{\alpha,\beta} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i[y_i(\mathbf{w}\cdot\mathbf{x_i} - b) - 1 + \xi_i] - \sum_{i=1}^{n}\beta_i\xi_i \right\}$ (e)

Table5.1. Equations of SVM for resolved issues

5.1.1.Application based SVM

SVM has been studied by various scholars in different respects. It has been used for various kinds of applications.

Several of them are mentioned here:-

S. No.	**Author**	**Concern**	**Objective**
1.	Bennett and Campbell	Geometrical Point of view	
2.	Hebrick[176]	Bayesian rule	exposition of kernel method
3.	Hastie, Tibshirani and Friedman		Statistical literature on SVM

Table 5.2 **Concerned method and objectives of studies**[177]

According to Marquez and Gimenez (2006), the SVM Tool is a simple and effective classifier and generator of sequential data based on Support Vector Machines. SVM tool is robust, flexible for feature modeling. It is vary suitable for the development of NLP application for practical use.

In the comparison of other available statistical tagger the process of automated tagging is very fast. There is less or no feature parameters are required for tuning[178].

[176] Yi lin, 1999, Support vector machine and the bayes rule in the classification, technical report no.1014, Department of Statistics University of Wisconsin.

[177] Javier M. Moguerza and Alberto Munoz, (2006)" Support Vector Machines with Applications", Statistical Science 2006, Vol. 21, No. 3, 322–336 Institute of Mathematical Statistics.

[178] Jes´us Gim´enez and Llu´ıs M`arquez., *SVMTtool:Technicalmanual v1.3,* August 2006

After being checked with rigorous experiments and evaluation, it can be said that taggers based on SVM are one of the most efficient for the annotation of text from any language. The requirement of the SVM is huge amount of Data.

So far as the accuracy is concerned, as put forward by Marquez and Gimenez (2006):

> "The SVM-based tagger significantly outperforms the TnT tagger exactly under the same conditions, and achieves a very competitive accuracy of 97.2% for English on the Wall Street Journal corpus, which is comparable to the best taggers reported up to date. This version is implemented in Perl. A most efficient C++ version is currently available. The SVM light software implementation of Vapnik's Support Vector Machine (Vapnik, 1995) by Thorsten Joachims has been used to train the models".

In machine learning, support vector machines by Vapnik, as cited in Joachims (1999), are supervised learning models with associated learning algorithms that analyze data and recognize patterns. They are applied for classification and regression analysis.

If a set of training examples is provided, by marking each of them as belonging to some of the categories, an SVM training algorithm prepares a model that labels tags to new input examples, which makes it a 'non-probabilistic binary linear classifier' (Marquez and Gimenez, 2006).

5.1.2. Property of the Tool

The tagger is said to have the following properties, as mentioned in Giménez and Màrquez (2006) :

- The tool has very simple configuration and installation procedure. Training of the tagger is comparatively easy and has very few parameters to fix.
- It is flexible at defining wider feature pattern and justifies with the shape and size of the context.

- Given a set of N training examples $\{(x_1, y_1),\ldots, (x_N, y_N)\}$ where every instance x_i stands for a vector R_N and class label is $y_i \in \{-1,+1\}$. An SVM learns a linear hyperplane that separates the set of positive examples from the set of negative examples with maximal margin; the margin is defined as the distance of the hyperplane to the nearest of the positive and negative examples (Marquez and Gimenez, 2006)

- It is a robust machine which allows sentence level analysis and other strategies. The soft margin learning algorithm by tuning the parameter C is introduced. The soft margin classification utilizes the following equation

  $$\{ z : \langle w, z\rangle + b = 0 \}$$

  Soft margin can be understood as a variant of optimizations problem where the parameter C is used to balance the margin maximization and errors occurred in training

- The ability to learn with very limited language data accounts for its portability. It is a semi-supervised machine and does not require full-fledged knowledge of language prior to the training.

- SVM can work accurately in indefinite feature space with similar level of accuracy.

- The use of linear kernel helps accelerating the tagging speed of the present Perl prototype tagger to 1500 words per second and the C++ version with the speed of 10,000 words per second.

- The linear separator is defined by two components: a weight vector w (with one component for each feature), and a bias b which stands for the distance of the hyperplane to the origin. The classification rule of an SVM is:

  $\mathrm{sgn}\ (f\ (x, w, b))$ (1)

  $f\ (x, w, b) = \langle w \cdot x\rangle + b$ (2)

  being x the example to be classified. In the linearly separable case, learning the maximal margin hyperplane (w, b) can be stated as a convex quadratic optimization problem with

a unique solution: minimize ||w||, subject to the constraints (one for each training example):

yi (<w · xi> + b) ≥ 1 (3)

Given some training data **D** , a set of n points of the form

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{R}^p,\, y_i \in \{-1, 1\}\}_{i=1}^{n}$$

So, they have developed the system using SVM (Joachims, 1999), which performs classification by constructing N-dimensional hyperplane that optimally separates data into two categories.

5.1.3. Design of the tagger

The tagger has been designed to have three main components SVMTlearn, SVMTagger and SVMTeval. These components responsible for creating model file for the training data, tagging the test files and evaluation of the results obtained by the tagger, respectively.

5.1.4. Tagger Models

So far, the SVM has been tested for five different models with slight but noticeable differences. The models implemented, are named 0 to 5 which stands for the features contained in them (Giménez, 2006). These models are as follows:

1. *Model 0-* This is the default model trained on one pass scheme. This is a unidirectional tagger which tags either in left to right or right to left direction. Being a one pass, the tokens not disambiguated in advance remains ambiguous.

2. *Model 1-* This disambiguates the unseen context of the previous text. Hence, known as the second pass schema.

3. *Model 2-* The POS features are not the concern of this model. Working on the one pass it calls for the second pass to review the tagging results.

4. *Model 3-* This considers the unambiguous words from the annotated corpus for training the tagger.
5. *Model 4-* This handles the errors caused by the unknown words. This is done by creating different folders and generating dictionary. Before tagging the tagger looks into the dictionary and the words found in any folder but not in the rest of the corpus are marked unknown.

5.2. Experimental Set-ups for SVM

This is one of the prominent sections computationally as it encapsulates feature extraction, configuration files, training, testing and evaluation data and its format for both the models.

5.2.1. Feature Extraction

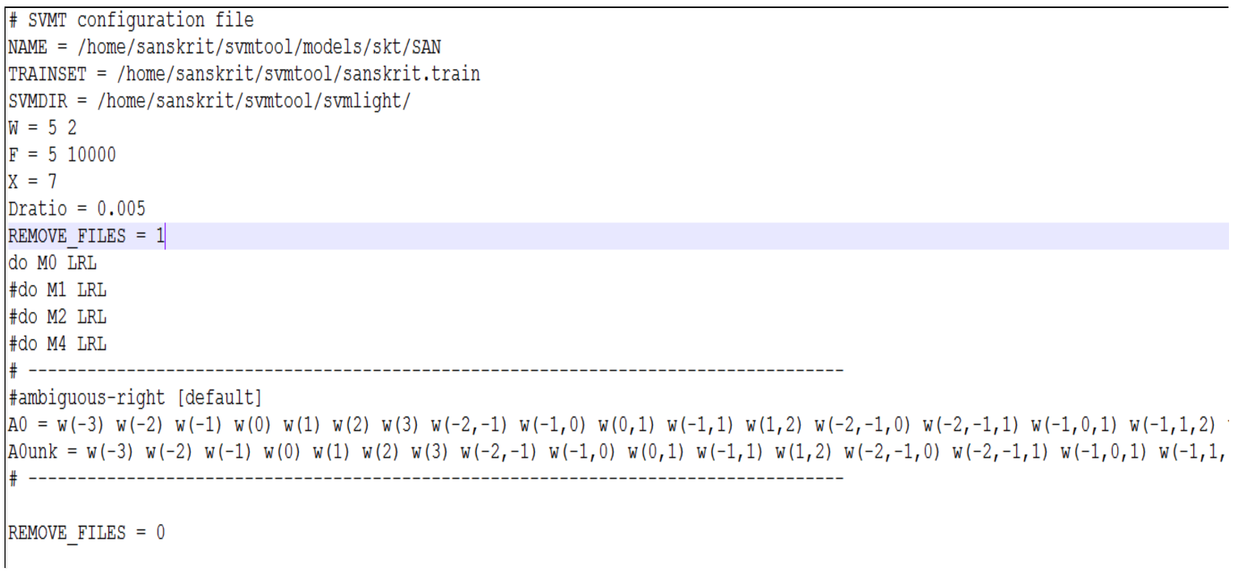
For both the models, simple features have been selected and several other features have been set to default mode. So far as the former is concerned, learning phase contains medium verbose (-V 2) and the mode of learning and tagging is set to left-right-left (LRL). The rest of the features like sliding window, feature set, feature filtering, model compression, C parameter tuning, Dictionary repairing and so on are set to the default mode.[179]

5.2.2. Configuration

There are many options like verbose, sliding windows and feature sets etc. For more on this please visit Giménez[180]. The verbose throughout the experiment, was set as medium throughout the experiment. The tagger was set to perform a two pass, unidirectional tagging from right to left.

The tagger employs different tagging strategies at different tagging levels like strategy for running in one or two pass, choosing the direction for the tagger, filtering out thresholds for known and unknown tokens, prediction of POS to be tagged, backup lexicon, lemma lexicon etc as part of internal process done by the tagger, as the tagging continues. And the rest of the features like sliding window, feature set, feature filtering, model compression, C parameter tuning, dictionary repairing and so on have been set to the default mode. The following feature template has been configured for the known and unknown ambiguous words.

[179179] http://taku910.github.io/CRF++pp/
[180] ibid



```
# SVMT configuration file
NAME = /home/sanskrit/svmtool/models/skt/SAN
TRAINSET = /home/sanskrit/svmtool/sanskrit.train
SVMDIR = /home/sanskrit/svmtool/svmlight/
W = 5 2
F = 5 10000
X = 7
Dratio = 0.005
REMOVE_FILES = 1
do M0 LRL
#do M1 LRL
#do M2 LRL
#do M4 LRL
# ---------------------------------------------------------------------------------
#ambiguous-right [default]
A0 = w(-3) w(-2) w(-1) w(0) w(1) w(2) w(3) w(-2,-1) w(-1,0) w(0,1) w(-1,1) w(1,2) w(-2,-1,0) w(-2,-1,1) w(-1,0,1) w(-1,1,2)
A0unk = w(-3) w(-2) w(-1) w(0) w(1) w(2) w(3) w(-2,-1) w(-1,0) w(0,1) w(-1,1) w(1,2) w(-2,-1,0) w(-2,-1,1) w(-1,0,1) w(-1,1,
# ---------------------------------------------------------------------------------

REMOVE_FILES = 0
```

Fig.3. Configuration file of SVM Tagger

5.3. Training the SVM Model

A set of annotated or un-annotated examples, is given to the SVM model, then it trains the set of SVM classifiers. For training o those classifiers, the SVM models use SVM-light7. SVM-light7 is an implication of Vapnik's SVMs in C and developed by Thoresten Joachims. The SVM light software implementation of Vapnik's Support Vector Machine in 1995 by Thorsten Joachims has been used to train the models (Joachims, 1999).

For training, around 155k data have been used. During training, it is important that the training data is in put in columns. The columned manner data should be i.e. 'a token per line corpus', and in a sentence-sentence fashion. The very first field needs to represent the token while the second one is the corresponding tag. The rest of the columns are not so necessary for parts of speech tagging and can contain additional information (Marquez and Gimenenez, 2006). For example

SVMTlearn behavior is easily adjusted through a configuration file. These are the currently available options:

- ❖ Sliding window (size and core position)
- ❖ Feature set (word features, POS features, orthographic features,
- ❖ 'multiple-column' features)

- ❖ Feature filtering (count cutoff and max mapping size)
- ❖ SVM model compression
- ❖ C parameter tuning
- ❖ Test [against a test set or via cross-validation]
- ❖ Dictionary repairing (either heuristically and/or based on a correction
- ❖ list)
- ❖ Ambiguous classes (may be optionally provided)
- ❖ Open classes (may be optionally provided)

Backup lexicon (may be optionally provided)

5.3.1. Cautions for the training file

Several kinds of errors might remain in the training file after validation and formatting. One has to be cautious about the proper formatting of the data. Some common errors in the file might prove to be a nuisance during the training process. Some of them encountered in due course are shared below:

- There should be no numbering before the word tokens.
- No blank line or extra space should be left in between two line or words/tokens.
- A proper one to one, parallel alignment of words and tags should be there. A system failure can occur if the tagger finds either of the elements missing in a single line.
- During training, there should not be any extra information and columns except the two (for word and token).

5.4. Testing of the Tagger

SVMTagger module is synonymous to testing. According to Gimenenez and Marquez (2006), SVMTagger annotates the part of speech of a text, when it gets the path of a SVM model file which is previously learned. It also includes the dictionary, which is generated automatically during training. The SVM standard i.e. one token per line must be followed by the input file.After the successful training of the model file, the tagger was set at the task of tagging the test files. The testing is done in two phases- initial testing and currently running. Testing was done on seen and unseen data for both the test phrase.

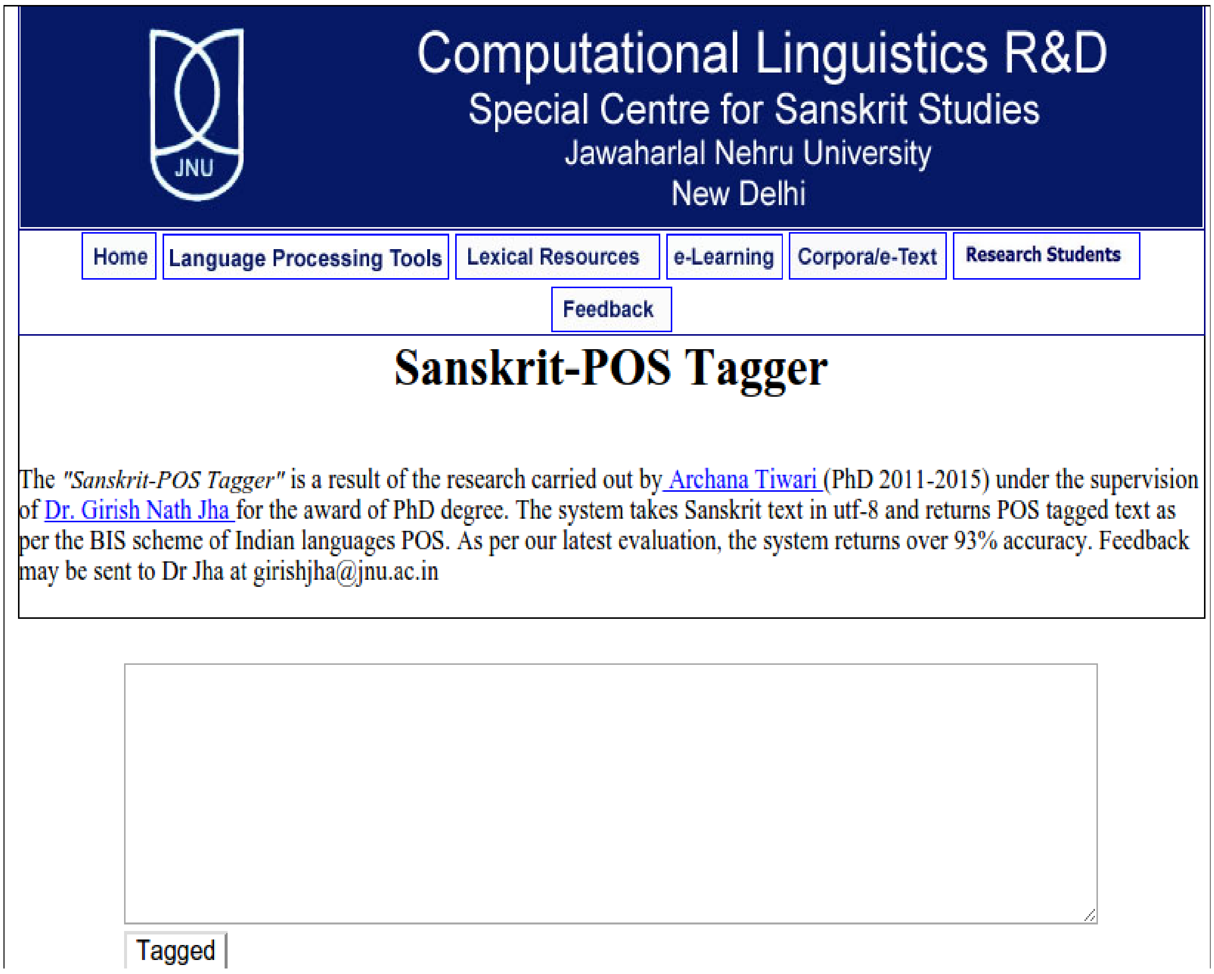
The automated annotation process takes place on-line showing a sliding window which presents feature context to be selected at every decision by the tagger. The tagger performs the parts of speech tagging in a sentence by sentence fashion and a standard input/output system; with the token is expected to be the first column and the tag to be the second one.

These are some of the currently available options for the user.

- ❖ Tagging scheme (greedy/sentence-level)
- ❖ Tagging direction (left-to-right, right-to-left, or both)
- ❖ One pass / Two passes
- ❖ SVM Model Compression
- ❖ Get all predictions (not only the winner)
- ❖ Use of a softmax function to transform predictions into probabilities
- ❖ Backup lexicon (may be optionally provided)
- ❖ Lemma lexicon (may be optionally provided)
- ❖ Use of a Levenshtein Distance module to enrich the dictionary taking in
- ❖ account the input

5.5. The Online User Interface of the Sanskrit Statistical Tagger

The tool is a web-based platform which is implemented with the JSP code and run with Apache Tomcat on the server. It will be available online on the web of the Computational Linguistics Research and Development, SCSS, JNU with the link (http://sanskrit.jnu.ac.in/pos/sanskrit.jsp). The online SVM/CRF++ Taggers for Sanskrit has the following interactive user interface structure.


Computational Linguistics R&D
Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New Delhi
JNU
Home
Language Processing Tools
Lexical Resources
e-Learning
Corpora/e-Text
Research Students
Feedback
Tagged

Sanskrit-POS Tagger

The *"Sanskrit-POS Tagger"* is a result of the research carried out by Archana Tiwari (PhD 2011-2015) under the supervision of Dr. Girish Nath Jha for the award of PhD degree. The system takes Sanskrit text in utf-8 and returns POS tagged text as per the BIS scheme of Indian languages POS. As per our latest evaluation, the system returns over 93% accuracy. Feedback may be sent to Dr Jha at girishjha@jnu.ac.in

Screenshot 1. The User Interface of Sanskrit POS Tagger

5.6. Architecture for Sanskrit Tagger

The architecture of the tagger developed in present research is given shown here:-

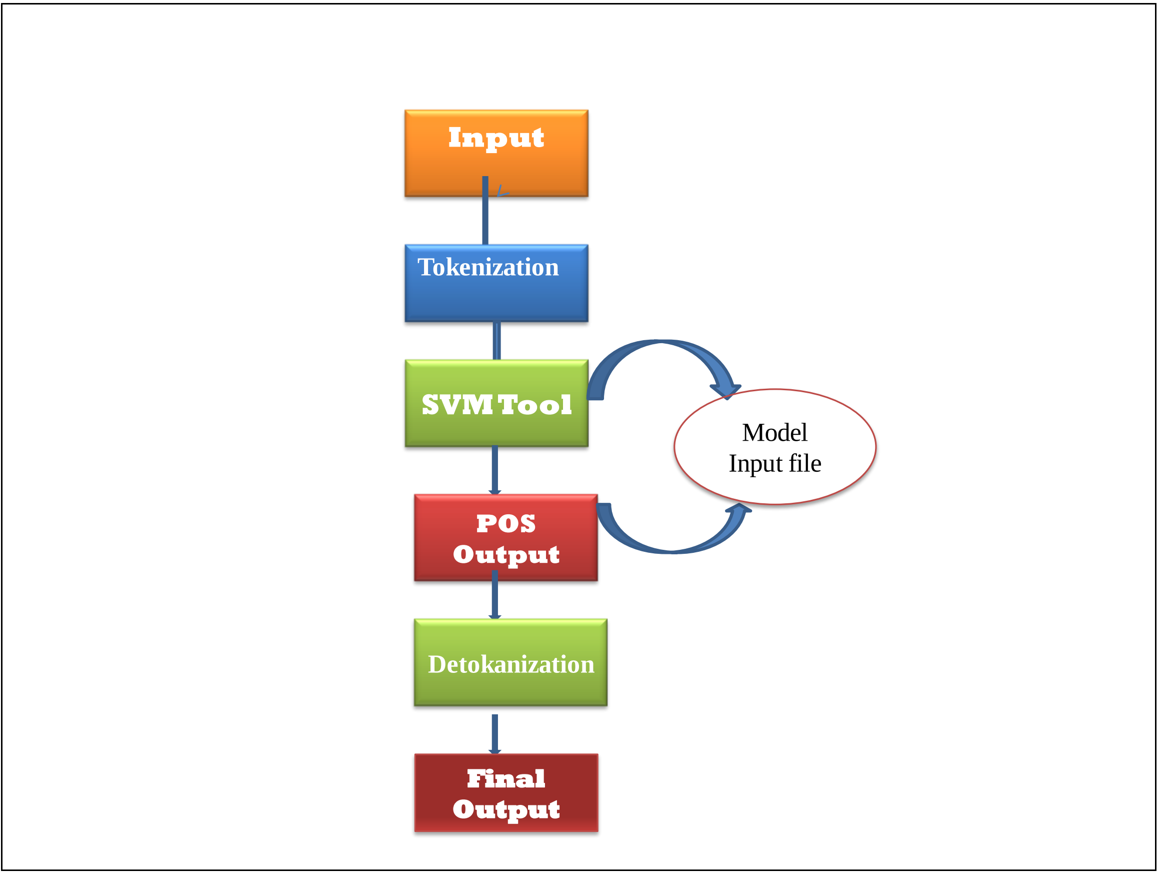


Fig. 5.3. Architecture of The Sanskrit Tagger

5.6.1 Input Text

Firstly, a user provides the input files or sentences in Devanagari script to the online platform. The platform identifies only the UTF encoding of the raw input text to be processed. If a user encodes the input file with editor software other than the Unicode font it will not be identifiable by the tool. However, there is no specific limit in terms of the quantity of the data to be given. There is no lower limit of the data to be encoded; it can be a single token.

5.6.2. Pre-Processor

The pre-processor filters the input text and checks whether any unwanted components are not present in the same. If it finds out so, it either discards them from the input text or leaves as they were earlier during the input. For example, if it finds non-specified characters like the unwanted punctuations within the token or half finished letters or any other 'control characters' (Choudhary, 2006), it leaves them as they are by labeling with a tag.

Input token:

शङ्करः एकं फलं स्वहस्तेन् अगृह्णात् ।

Output token :

शङ्करः\N_NNP एकं\QT_QTC फलं\N_NN स्वहस्तेन्\N_NN अगृह्णात्\V_VM_VF ।\RD_PUNC

5.6.3. Tokenization

The next step that the tool approaches to is that it tokenizes the input data which is encoded in a sentence-by-sentence fashion. Further, it tokenizes the input data wherever it finds two tokens separated by a white space. Thereafter, it converts the file with sentences to token-by-token fashion. The tokenizer used in the tool is the Java Class Tokenizer.

5.6.4. The SVM Tool/CRF++ Toolkit

Thirdly, the Tool forwards to the SVM tool Toolkit which is run by the SVM algorithm. It reaches to the model and input files and implements them. At this important stage, the SVM processes the input data in two phases: the LR mode and the RL mode. Thereafter, it annotates them based on its previous learning and provides the output identifying the probable tag for the given input token. If one selects the SVM tagger to process, the SVM tool will annotate the input file. If one selects the CRF button, the toolkit starts processing the data based on its earlier training.

5.6.5. The POS-tagged Output

The efficiency of the training data decides the quality of the output decoded by the tagger. To make the tagger more efficient, one needs to focus much on the training period. The output generated by the tagger is in a token-by-token fashion in each line. It solely depends upon the input file as to what will be the probable best output of the input data. If a user provides a phrase, the tagger provides the tagged phrase by tokenizing it, provided there are no punctuations or any control character in between or attached with the token.

5.6.6. The De-tokenizer

The tokenizer tokenizes each linguistic element into token while the de-tokenizer detokenizes them into the previous order. So the tokenizer and the de-tokenizer are contrary to each other. Thus, the de-tokenizer converts the tagged output text into its tokenized forms; separating each token and tag with a white-space. Thereafter, the tool provides the final output.

5.7. Technology Used for Making the Tool

The front end data for the application has been developed applying Servlets, Java Server Page, and HTML. The JSP page has been UTF-8 enabled and supports the scripts of any language encoded with UTF-8. The online platform runs on the Apache Tomcat 4.0 which is a container Java Servlet and Java Server Pages and the back end data of the tool.The data opens online in a web browser which is based locally on the user’s computer. The URL opens the JSP file located on the host computer usually at the path given. The browser, with the help of the java-webserver, reads the sanskritsvm.jsp file. To understand the structure and functions of the said file, one needs to have a look at the following.

The technical environment of the application is as follows:

- Programming language used is Java
- Web-based Tools are Servlets and JSP
- Server used is Apache Tomcat 4.0

5.7.1. Apache Tomcat 4.0

Apache Tomcat 4.0 supports the web applications that are built for the Servlet 2.2 and JSP 1.1 specifications (applied with no change) which was officially announced on September 17.66 This is developed in an open and participatory environment and released under Apache Software License. Tomcat is a web-based server software for developing and running Java server pages on a local host (Chowdhary, 2006).

5.7.2. JSP

Java server pages are html pages which use Java objects embedded in the html code. JSP technology is an extension of the servlet technology created to support authoring of HTML and XML pages. It makes the process easier to combine fixed or static template data with dynamic content. Even if one is comfortable in writing servlets, there are several compelling reasons to investigate JSP technology as a complement to their existing work. Java Server Pages are utilized in creating webpage content with the application of Java-written XML and scriplets67.

5.7.3. Java Servlet Technology

A servlet is an application program, written in Java and executed on a java compatible web server. It is applied for enhancing and extending the Web servers. One of the reasons for it being user-friendly is that it is ‘server and platform-independent’68. It can avail all the benefits of Java language like portability, performance, reusability, and protection.

“A reference to a servlet appears in the mark-up for a web page, in the same way that a reference to a graphics file appears. The web-server executes the servlet and sends the results of the execution (if there are any) to the web browser as HTML text” (Chowdhary, 2006).

5.8. Evaluation of the Tagger

The evaluation tool provides different sets of words: known vs unknown, ambiguous vs unambiguous, ambiguous known words vs unambiguous known words. Further, it provides lists of words commonly ‘sharing the same degree of ambiguity’ with various levels of ambiguity classes. These features are created automatically based on the morphological dictionary that it generates during training time. It provides the evaluation results in the following lines.

A brief report on overall accuracy

- A comparison of known vs unknown words and ambiguous and unambiguous words
- Grouping of words as per their level of disambiguation complexity
- Grouping of words as per their class of ambiguity
- Presentation of the accuracy from the perspective of parts of speech

For the inspection of the type of error and patterns of errors was done. The validation of the data tagged by tagger was done by the researcher. During evaluation process similar description is updated. But they only gave access to the nature of ambiguity and its classes. The benefit of manual validation is that it will bring researcher directly to the issue and the nature of the problem.

According to Gimenenez and Marquez (2006), the SVMTeval evaluates the performance of the statistical tagger in terms of accuracy, if provided the predicted tagging output of the tagger and its corresponding gold standard data. It is a quite useful feature for "the tuning of the system parameters, like the C parameter, the feature patterns and filtering, the model compression et cetera".

5.8.1. Gold corpus

First of all, the gold corpus is created out of the training set. The gold corpus is a thoroughly revised training data which account for 100% accuracy (ideal state). The gold corpus created in this experiment, went through two fold validation of the training sets and evaluated in the following stages:

1. Test result evaluated for seen data (part of training set) in the initial phase (eval A, hereafter).

2. Test result evaluated for unseen data (other than training set) in the initial phase (eval B, hereafter).

3. Test result evaluated for seen data (part of training set) in the current phase (eval C, hereafter).

4. Test result evaluated for unseen data (other than training set) in the current phase (eval D, hereafter).

5.8.2. SVMTeval

The next module 'SVMTeval' stands for the evaluation of the tagger generated output. As discussed by Gimenenez and Marquez (2006), the SVMTeval evaluates the performance of the statistical tagger in terms of accuracy, if provided the predicted tagging output of the tagger and its corresponding gold standard data. It is a quite useful feature for "the tuning of the system parameters, like the C parameter, the feature patterns and filtering, the model compression et cetera". The evaluation tool provides different sets of words: known vs unknown, ambiguous vs unambiguous, ambiguous known words vs unambiguous known words. Further, it provides lists of words commonly 'sharing the same degree of ambiguity' with various levels of ambiguity classes. These features are created automatically based on the morphological dictionary that it generates during training time. It provides the evaluation results in the following lines.

- A brief report on overall accuracy
- A comparison of known vs unknown words and ambiguous and unambiguous
- words
- Grouping of words as per their level of disambiguation complexity
- Grouping of words as per their class of ambiguity
- Presentation of the accuracy from the perspective of parts of speech**.**

These are as follows:

a) The first section is a simple listing of different kinds of tokens including known, unknown, and ambiguous and their accuracies followed by the total no. of hits and trials for each known and unknown class of tokens.

Per Level of ambiguity'

b) The second section is the accuracy calculated 'per level of ambiguity'. The tagger, here, constructs classes for levels of ambiguity depending upon the hits and trials attempted by the tagger. These witnessed a shift in the ambiguity ratio as moving from one level to another. The ambiguity as per the level, postulated 6 classes for both the sets tested in the current phase.

Accuracy per class of ambiguity'

c) Next section is about 'accuracy per class of ambiguity'. This entails a detailed classification of the ambiguous tags. There are 213 and 306 classes postulated by the tool for unseen and seen test set, respectively.

'Accuracy per Parts-of Speech'

d) This second last section is labelled as 'accuracy per Parts-of Speech'. This gives a description of average accuracy acquired by tagger for each and every tag.

e) The last section is the announcement of the overall accuracy achieved by the tool for the present test sets.

Tagger generated evaluation report has been enlisted in the appendices, at the end of the dissertation.

5.8.3. Overall Evaluation

Results are always compared to the most-frequent-tag (MFT) baseline (Giminenez and Marquez, 2006). The results are based on the automatic evaluation of the SVM Tagger for Sanskrit by the SVMTeval tool. The following figure demonstrates that the first bar which stands for the known words has an accuracy rate of 82. While the bar of unknown words states that it is around 17.26%. The reason for low accuracy is that, Hits of the unknown words is less in number. The accuracy rate of the ambiguous words out of the total number of the evaluated data is 17.34%. The MFT baseline is 78.9%. Out of the total number of ambiguous tokens (17.34%), known ambiguous are 78.9% whereas the unambiguous known tokens account for 99.77%.

```
EVALUATING </home/sanskrit/svmtool/skttest.out> vs. </home/sanskrit/svmtool/golds> on
model </home/sanskrit/svmtool/models/skt/SAN>...

..........10000..........20000..........30000..........40000..........50000....51522 tokens [DONE]

* ================= TAGGING SUMMARY
=========================================================

#TOKENS          = 51522

AVERAGE_AMBIGUITY = 7.8028 tags per token

* -----------------------------------------------------------------------------------

#KNOWN           = 82.7394% -->          42629 / 51522

#UNKNOWN         = 17.2606% -->          8893 / 51522

#AMBIGUOUS       = 17.3402% -->          8934 / 51522

#MFT baseline    = 78.9003% -->          40651 / 51522
```

KNOWN vs UNKNOWN TOKENS

	HITS	TRIALS	ACCURACY
KNOWN	41254	42629	96.7745%
Known Unambiguous Tokens	33612	33695	99.7537%
known ambiguous tokens	7642	8934	85.5384%
Unknown	6770	8893	76.1272%

Table5.3. Table of known and unknown tokens

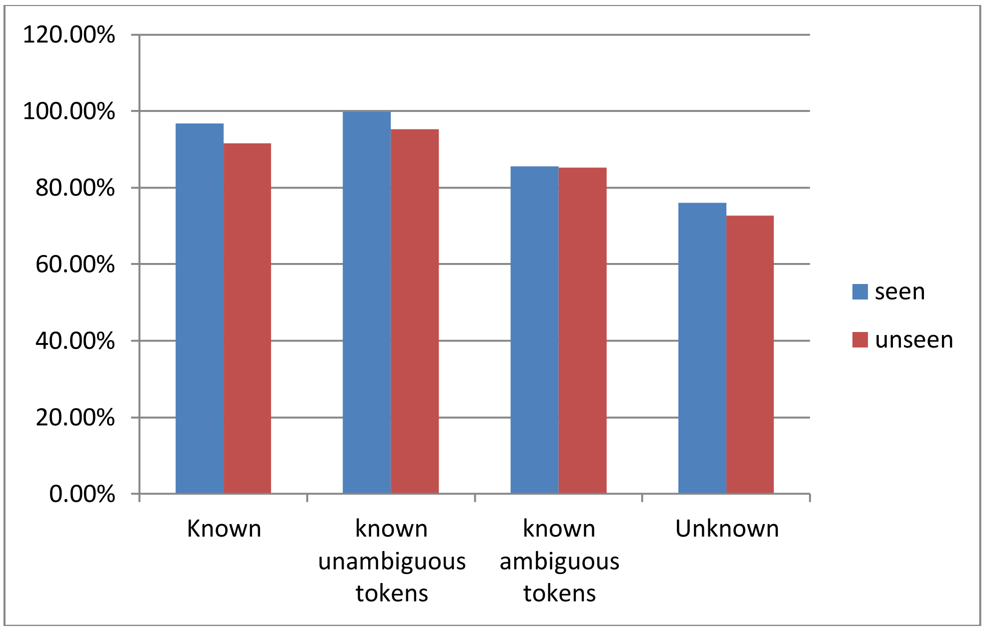


Chart.1 Result Summary of the Odia SVM Tagger on the Seen and Unseen Data

5.8.3.1. Accuracy per Level of Ambiguity

The following tabulated data represents the class of ambiguity and along with their number of hits and trails, accuracy per level and MFT baseline during mechanical evaluation. There are nine ambiguity classes prepared by the machine. The highest ambiguous level that one of the levels has is the eighth one which includes some the ambiguity sets having the lowest accuracy rates. On the other hand, lowest ambiguous class is the first level.

Level	Hits	Trials	Accuracy	MFT
1	33612	33695	99.7537%	99.8783%
2	5322	6187	86.0191%	81.7682%
3	1617	1901	85.0605%	73.1194%
4	517	603	85.7380%	75.4561%
5	161	207	77.7778%	36.2319%
6	16	23	69.5652%	47.8261%
8	9	13	0.00%	0.0000%
39	6770	8884	76.2044%	0.0000%

Table.5 Accuracy rate per level of SVM tagger

5.8.3.2. Accuracy as per class of ambiguity

For SVM tool, the ambiguity reported by the tagger is inversely proportional to the complexity in morphology. Indian languages are rich in morphology as compared to English. Giménez and Màrquez (2006) also confirms to the hypothesis that morphologically rich languages are found to have lesser degree of ambiguity as the SVM Tool trained as part of their study claimed 93.91% ambiguity for Wall Street Journal, corpus of English and 95.04% for LEXESP, corpus of Spanish language.

The ambiguity classes are formed considering the possible tags given to each word token. A list of all unique ambiguity classes can be found in appendices on the basis of which these classes are further classified into the following categories:

(i) *Major classes:* The major class in this categorization includes all unique classes of ambiguity found in all the evaluation reports generated by the tagger. At present , there are 358 major classes. A descriptive list of all ambiguity classes are given in the appendix

(ii) *Classes with erroneous tags:* The annotated corpus used during training is a semi-automatically tagged corpus, which is prepared with the help of manual validation. This is quite possible that there have some typo errors like misspelled tags, J for JJ i.e. tag for adjective. There are only two such inconsistencies found in the data. They are –

CQT- Quantitative- 8

N_N –Common Noun-14

NN- N_NN-common noun

N_NNPP- N_NNP- Proper noun=10 occurrences with a total of only 12 occurrences in both the tested data sets.

(iii) *Classes with single tag:* All tags as presented in the POS tagset come in this class.

(iv) *Classes with two tags*: This includes those words which have at least two possible tags. Let say, any word which functions both as a noun and an adjective will fall in this category. For example- *JJ_N_NNP*

(v) *Classes with three tags*: The words, which have three possible tags, are listed under this category. For example, *JJ_N_NN_RB*

(vi) *Classes with more than three tags:* This category ranges up to maximum eight possible function tags for a given word/token as listed by the tagger. Though, no such situation noticed during manual annotation. Up to four usages can be accounted from the corpus

5.8.3.3. Accuracy per POS Category of the SVM Tagger

The following data represents the accuracy per level of parts of speech for the Sanskrit SVM statistical tagger.

In the evaluation of the tagger, of all the POS categories, Negation has the highest rate of accuracy based on the number of hits. Succeeding negation, those categories which registered highest accuracy are Quotative conjunction, punctuation class, gerund verb, common noun and so on. The lowest accuracy based on hits is registered in the class of echo words as it gets only 3 hits, although it has 19 trials in gold data. Reciprocal Pronoun is the second class which has accuracy less than 50 %. The overall accuracy rate of the tagger for Sanskrit is 93.21% with a baseline MFT of 78.90%. This is the highest accuracy achieved by any tagger of Sanskrit till now.

Following table contains accuracy of all the POS category based on HITs and Trials.

POS	**HITS**	**TRIALS**	**ACCURACY**	**MFT**
CC_CCD	1353	1589	85.1479%	18.3134%
CC_CCS	361	473	76.3214%	57.2939%
CC_CCS_UT	722	724	99.7238%	4.2818%
DM_DMD	200	304	65.7895%	60.1974%
DM_DMQ	1	1	100.0000%	100.0000%
DM_DMR	14	28	50.0000%	10.7143%
JJ	490	641	76.4431%	84.0874%
N_NN	23005	23517	97.8229%	93.7535%
N_NNP	1714	2022	84.7676%	81.9980%
N_NNPP	11	11	100.0000%	81.8182%

N_NNV	53	85	62.3529%	60.0000%
N_NST	114	141	80.8511%	46.0993%
PR_PRC	4	11	36.3636%	36.3636%
PR_PRF	70	81	86.4198%	75.3086%
PR_PRL	196	257	76.2646%	40.4669%
PR_PRP	2368	2575	91.9612%	64.0388%
PR_PRQ	342	384	89.0625%	46.6146%
PSP	124	137	90.5109%	59.8540%
QT_QTC	751	851	88.2491%	78.0259%
QT_QTF	67	159	42.1384%	22.0126%
QT_QTO	140	166	84.3373%	80.7229%
RB	114	136	83.8235%	44.1176%
RD_ECH	3	19	15.7895%	10.5263%
RD_INTF	2	2	100.0000%	100.0000%
RD_PUNC	5475	5858	93.4619%	84.8242%
RD_RDF	167	186	89.7849%	88.1720%
RD_SYM	799	943	84.7296%	38.4942%
RD_UNK	1	1	100.0000%	100.0000%
RP_INJ	103	115	89.5652%	38.2609%
RP_INTF	59	61	96.7213%	57.3770%
RP_NEG	816	818	99.7555%	4.5232%
RP_RPD	1841	2161	85.1920%	39.2874%
V_VAUX	52	60	86.6667%	81.6667%
V_VM	13	19	68.4211%	47.3684%
V_VM_VF	5370	5685	94.4591%	88.0739%
V_VM_VINF	223	262	85.1145%	66.4122%
V_VM_VNF	203	276	73.5507%	64.1304%
VM_VNG	636	672	94.6429%	89.1369%
Total	48024	51522	93.2107%	78.9003%

Table.6. Accuracy Rate per POS Category of the SVM Tagger

5.9. Error Analysis

In this analysis some of the terminologies have been taken from Manning (2011). The error committed by tagger needs to thoroughly reviewed, for increasing the efficiency and accuracy of the tagger. In this work, during identification of the errors, it has been found out that the tagger is committing some overlapping errors. A pie chart has been given that demonstrate the error-categories, where one need to focus at. The errors are categorized in following broad categories :

5.9.1. Open-class Categories

In categories of words or part of speech, there are two basic classes : Open class and close class categories. Those words, which can change over the time and add some new word to the lexicon, come under open class. The close class, on the other hand are those part-of speeches which has a fix character and there is very less or no chance for expansion in their nature in lexicon.The nouns, verbs and adjectives are examples of open-class, while the close class are the fixed function words such as prepositions or postpositions, adverbs, demonstratives, pronouns, conjunctions, particles, determiners, symbols, punctuations etc. The open-class foreign words can add to the lexicon by enculturation, getting transliterated or used Sanskrit-like.

For example

Kāśmīraṃ \N_NN ('kāśmīraṃ' a proper noun)

śītakālīnā\N_NN (the 'winter' is an adjective)

5.9.2. Unknown Words

When a word does not appears for even a single time in the training lexicon then that word is an unknown word. Some time these words are from the same language or foreign language words transliterated into Sanskrit. Sanskrit has a tradition of using long compound words, because of that the tagger is not able to recognize a word's category. The known words have already registered their presence at least for once in the training data.

For example

Devadānavagandharvayakṣarakṣomahoragaiḥ \N_NN

This is an example of a compound word which never occurred in the training data.

5.9.3. Lexicon Gap

When a word came several time with a specific tag in training data, but during evaluation the tagger gives that word a different tag.

For example

Training data token

Himālayaḥ/N_NNP *api*/CC_CCD *sainikaḥ*/N_NN *iva*/RP_RPD *asya*/RR_PRP *rakṣāṃ*/N_NN *karoti*/VM_VM_VF

Evaluation data token

Himālayaḥ/N_NN *api*/CC_CCD *sainikaḥ*/N_NN *iva*/RP_RPD *asya*/RR_PRP *rakṣāṃ*/N_NN *karoti/*VM_VM_VF

In the above example, the only tag that the word / *Himālayaḥ/* gets in all the cases is N_NNP, but during evaluation, the wrong tag labeled by the tagger is common noun N_NN.

5.9.4. Difficult Linguistics

Some time there are words which has ambiguous and problematic tags. These tags are hard to decide even for the human annotator. When the tagger tags them it commits mistake and gives a wrong tag. These types of cases pertain to the other disciplines of linguistics like syntax, semantics and discourse than morphology. In Sanskrit, it is quite difficult to judge in the conjunct verb constructions (JJ/N_NN/N_NNP+V_VM_VF) whether the first lexical component is noun or adjective in several cases. Similarly, the cases of adverbs, demonstratives etc. are the other cases.

For Example

***Kuśalaḥ*/N_NN** *paṭhati*/V_VM_VF

In the above example ***Kuśalaḥ*** can be a proper noun or adjective.

5.9.5. Under-specified Labels

Those words which have more than two tags in training data are the unclear, ambiguous or under-specified words. Contextually unclear or undeterminable are also come in this category. Unclear, ambiguous or under-specified words are the words having more than one tags in the whole training corpus or contextually unclear or undeterminable. Ambiguous words can be of both the known and the unknown words. In Sanskrit, the average ambiguity for the SVM tagger is 17.2606 tags per token in the unseen data and 7.8028 tags per token in the seen data . /*bahu/* is a word having three tags which create ambiguity during the processing of evaluation data. Some words have other linguistic ambiguities like lexical, syntactic, semantic etc.

For Example

bahu/ (QT_QTF or RP_INTF or RB)

5.9.6. Inconsistent Gold Standard

Data inconsistency occurs when there are some cases where making a proper judgment is not possible and there is a disagreement between annotators about the category of the word. Because of this a situation occurs where different annotator tag those words based on their linguistic knowledge. This makes the data inconsistent. Because of the inconsistency of both gold training annotated data, the evaluated data becomes error-prone.

In the data explained below, /sahasraśo/ and /*asmin* / in both the training and gold file have been tagged inconsistently which results in an inconsistent output.

For Example

Vaidikāni/N_NN yoga/N_NN *viṣayāṇi*/N_NN ***sahasraśa*/QT_QTF**
upalabhyante/V_VM_VF |/RD_PUNC

***Sahasraśo*/N_NN** *janāḥ*/N_NN *kaṣṭa*/N_NN *anubhavanti*/V_VM_VF |/RD_PUNC

5.9.7. Wrong Gold Data

This kind of error generate when the gold file is wrongly tagged. Because of that the evaluation data also becomes wrong.For example, the /kṛpaṇaḥ/ in the gold data has been annotated as N_NN; because of that the evaluation data also become wrong.

saḥ/PR_PRP *atīva*/RP_INTF ***kṛpaṇaḥ*/N_NN** *āsīt*/V_VM_VF |/RD_PUNC

5.9.8. Multi-word Expressions

Lot of compound words occurs as multi words. Annotation of these words with a proper tag is really confusing even for human annotator. Because multi-word words contain some features that may not have the feature of the compound word. Those words get a general tag. There could be some modifiers appearing before the proper noun in a multi-word. In the following example, the word /*Samyukta*/ occurs as an adjective mostly. Therefore, this word in the following multi-word has been tagged wrongly as the adjective and the following word as a common noun that should be part of a proper named entity.

For Example

Samyukta\JJ *rashtra*\N_NN america\N_NNP 'United States of America'

5.9.9. Plausibly Correct

Some cases occurs when even if the data of training corpus and the gold file is correct, still the tagger is tagging those word inconsistently. These cases vary; some time the tagger is tagging them correct and sometime the given tag is incorrect. The context information of those word which are tagged wrongly, can be helpful for assigning the right tag. This can be possible if features are selected taking the consideration of the context of the word.

kiṃ/ PR_PRQ ***sarve*/ PR_PRP** *āropitā*/ N_NN *vṛkṣā* /N_NN *jīvitā* /V_VM_VF *vardhitāśca* /V_VM_VF । RD_PUNC

5.10. Suggested Solutions for the Statistical Tagger

In this section different approaches for making the tagger more accurate have been discussed. The statistical tagger prepared during this work is giving an accuracy of 93% which is highest accuracy for Sanskrit till now. But there is still scope of improvement in the quality, reliability and efficiency of the statistical tagger. For that several approaches can be used. In present work data approach has been applied and used. The other two approaches that can be used are linguistic rule formation and problematic part of speech and word sense disambiguation.

5.10.1.The Data Approach

The accuracy rate stage-wise shows that with the increase in the number of the tokens, the accuracy rate of the tagger increases. With each evaluation, results were evaluated and error analysis has been conducted manually. Based on the rule judgments of the human evaluator, corrections have been made. Initially, the accuracy rate has been evaluated manually, but the final evaluation has been performed by the machine. At the first stage with a training data of around 86k tokens the rate of accuracy was 87%, with 111k it was 89% and with 155k it rose up to 93.21% But, when it has been tested with the unseen data, the accuracy decreases to 76.12% because of a number of unknown and ambiguous words found by the tagger.

So it is possible that if the volume of Data increases the accuracy of the tagger is also going to increase.

5.10.2.Formulation of Linguistic Rules

Based on P*āṇ*ini’s grammar Sanskrit has a rich tradition of linguistic rules. It has well defined grammar. Using these rule for can be beneficial for

5.10.3.Adding lexicon

Sanskrit is a fix character because of a strict grammar, which is very near to formal grammar. Because of its static nature there is a very less possibility of the words to change. Adding a dictionary can be really useful for increasing accuracy of the tagger.

5.10.4.Pre-processing of Data

The extreme morphological and lexical richness of Sanskrit have a negative effect on the performance of the tagger. Therefore it is advisable that a pre-processing of data should be done before giving it to tagger for annotation.

- **Use of Morphological analyzer**

Sanskrit Language is of unique type with the extensive use of *samāsa* – compound words. Without proper analyzing of *samāsa*, it is not possible to understand a text. Therefore for decoding compound words use of morph-analyzer can be extremely useful. For example-

/Sukhamālikānugrahalabdhavratādeśaḥ/

The tagger will have difficulties in tagging this word as this kind of word seldom occurs in the corpora. But if the *samāsa* is decoded then the tagger will be more efficient in annotation of the component word of this word.

- **Use of *Saṃdhi* Splitter**

Use of *saṃdhi* splitter can be very helpful in resolving the euphonic rules called *saṃdhi* . This will resolve the problem of unknown words as a great length.

For example-

*Atilobhābhibhūtasya/*N_NN

The tagger has tagged the word as common noun. But if one uses the *saṃdhi* splitter then the word will have three tags-

Ati /RD_INTF

Lobha/N_NN

*Ābhibhūtasya/*N_NN

Because these words have multiple occurrences so the performance of the tagger will increase.

CONCLUSION

Introduction

POS tagging is a morpho-syntactic and a lexical problem and often the first phase in the development of a typical NLP application. Those words which have similar forms but have different categories cause the problem of ambiguity at different level. Sanskrit is morphologically and lexically a very rich language. A single word of Sanskrit is packed with lot of information. In Sanskrit, there is not a distinct demarcation between morphology, syntax and semantics. Because of this there is no clear distinction between the word-class categories. This feature of Sanskrit word-class categories makes their differentiation a complex task for the annotator. Problem of identification of nouns, adjectives and adverbs is a challenge to even scholars, making ways to multiple interpretations.

Summary of the Research

The first chapter of this work is a theoretical chapter, which gives an introductory description of the POS tagging and machine learning. In the first part of this chapter, concepts of tagging, advantage and classification of POS tagging are discussed. The second part of the chapter deals with machine learning. Different methods of machine learning, advantages and the rationale behind its use and various approaches are the topic of discussion.

The second chapter gives a brief history of POS tagging and survey of the taggers developed both in English and Indian languages. The first part of the chapter presents a short history of POS tagging and taggers developed in English. A detailed overview of Indian language taggers has been given in second part. Third part gives an illustration of taggers of less resource languages. Sanskrit taggers developed till date is given in the last part.

Feature of Sanskrit language are discussed in the third chapter. This chapter gives an insight into the grammar of Sanskrit. Aṣṭādhyāyī, the work of P*āṇ*ini, its components have been discussed in the first part. Further the chapter lays emphasis on Sanskrit morphology and gives details of its categories. Sanskrit and computation, Sanskrit sentence formation and a short history of POS in Sanskrit are given in brief. The problems of Sanskrit which create hurdles in preparation of language technology tools have given in the end of the chapter.

The fourth chapter deals with the topics related to tagset design, adopted tagset and preparation of corpora. Problems arising in deciding a particular tagset and its preparation related issues are given with a brief detail. The prominent tagsets like ILMT Tagset, Microsoft research India Limited (MSRI), Indian Languages Consortium (LDC) and Bureau of Indian Standard (BIS); are some tagsets whose features have been discussed here. The rationale behind the selection of BIS tagset, its features and issues and suggestion for improvement in the tagset is also given in detail. The last section of the chapter deals with the preparation of linguistic resources for Sanskrit in the BIS scheme. Problem of annotating Sanskrit, the stages of annotation in Sanskrit, process of corpora creation, validation and tokenization are discussed in the last part of the chapter.

This chapter is the most prominent chapter of this work as it gives the information about the tagger developed during the work. The chapter is divided in five sections. In the very first section a detailed description of the algorithm used in the research has been given. The reason behind the selection of SVM as training algorithm and its feature is given emphasis. The second section gives a detail about the experimental set-ups. In this part feature extraction, configuration, training, testing, and evaluation are the topic of discussion. The third section

The architecture for the user interface of the tagger is the given in the third section. The technologies used in development of the tool are given in brief in this section. The fifth and final section has the evaluation of the tagger, error analysis of the model are discussed. Proposed solution for the tool is given in the last.

Result of the study

In the present research, a tagger has been developed using statistical approach. The algorithm used in this works is Support Vector Machine (SVM). For training the tagger a linguistic resource has been created. First data is annotated using BIS tagset which is India's national standard. This hierarchical tagset has 11 categories. For further development of tagged corpora the bootstrap method was used. In this method at first, a relatively small amount of text is manually tagged and used to train a partially accurate tagger. Then the tagger is used for tagging data and afterwards that data is validated with caution. With this method around 155k token has been prepared for the training of the tagger.

The tagger was trained in three-fold training process with 86k, 111k, and 155k tokens. The tagger was evaluated with three-fold evaluation process. In the first evaluation the tagger was tested with 8133 tokens. The accuracy achieved by tagger was 87.3601%. The second evaluation was done using 12386 token and the accuracy was 89.0441%. In the third and final evaluation the tagger is evaluated with 51522 tokens. The accuracy of the tagger is

93.2107%. The MFT score of the tagger is 78.9003%. The unseen data, the accuracy decreases to 76.12%. The reason behind less accuracy in unseen data is huge number of unknown and ambiguous words found by the tagger.

Limitations of the system

There are several limitations of the statistical taggers in the present research work.

- The tagger's output was based on the validation of errors and customized training. There has been no other tools like NER, WSD, Morph Analyzer etc. has been used with the tagger. These tools can be helpful in increasing the efficiency of the tagger.
- The tagger has been prepared by only using data approach, where other approaches can be used for the increase of the accuracy of the tagger.
- The data was personally collected and tagged. The data lacks variety as only literature, current news and some blogs are used as the source of the data.
- Another important limitation of the research is that simple features selections for development of this tagger has been made and based on which results have been evaluated. The feature which has been selected were medium verbose (-V 2) and left-right-left (LRL) mode and used for the annotation process. The rest of the features were on default mode.

Issues and Challenges

Many issues were come during the development of the tagger. The human annotator and tagger, both faced challenges which are as follows:-

- High use of *saṃdhi* and *samāsa*, caused huge problem during annotation. Because of this the accuracy has affected at large scale.
- Lack of word order is also a challenging issue which caused problem. This feature is mainly occurred in the literary domain.
- As a classical language, poetic expressions of Sanskrit are hard to decode. This was one of the prominent issues for annotator and tagger.

Further Research and Development

The Tagger developed in this work can be used for many further R&D works. These are as follows:-

- This tool could be used for making chunker, parser, discourse anaphora resolution and machine translation platforms. Furthermore, ‘semi-supervised learning approaches’ can be applied to annotate automatically the freely available huge amount of unannotated data.
- Addition of ‘lexicon’ can increase the accuracy of the system.
- High use of saṃdhi and samāsa is a prominent feature of Sanskrit, which creates a lot of ambiguity issue and affects tagger’s accuracy adversely. Therefore use of saṃdhi splitter and morph-analyzer can be beneficial for increasing the efficiency and accuracy of the tagger.
- Application of tools like WSD, NER, Morph Analyzer, a suitable tokenizer, lexical database with prefix and suffix, dictionary look-up and post-processor can be useful for increasing the accuracy rate of the tagger.
- Accuracy rate can be increased by applying a contextual disambiguator and selecting better contextual features for the words and parts of speech labels.

BIBLIOGRAPHY

Bibliography

Articles

Aniket Dalal, Kumar Nagaraj, Uma Sawant and Sandeep Shelke, ***Hindi Part-of Speech Tagging and Chunking: A Maximum Entropy Approach***, In Proceeding of the NLPAI Machine Learning Competition, 2006

Antony P J, Santhanu P Mohan and Soman K P, (2010), ***SVM Based Parts Speech Tagger for Malayalam,*** International Conference on-Recent Trends in Information, Telecommunication and Computing (ITC 2010).

Antony P.J, Santhanu P Mohan, Soman K.P***,SVM Based Part of Speech Tagger for Malayalam***, IEEE International Conference on Recent Trends in Information, Telecommunication and Computing, pp. 339-341, 2010.

Atwell E, Demetriou G, Hughes J, Schiffrin A, Souter C, and Wilcock S, (2000). **'*A comparative evaluation of modern English corpus grammatical annotation schemes'.*** ICAME Journal, volume 24,pages 7-23, International Computer Archive of Modern and medieval English, Bergen

Behera Pitambar , Atul Kr. Ojha and Girish Nath Jha***,"Issues and Challenges in Developing Statistical POS Taggers for Sambalpuri"*** Centre for Linguistics, Jawaharlal Nehru University.

Bharathi Akshar, Prashanth R. Mannem, (2007), ***Introduction to the Shallow Parsing Contest for South Asian Languages***, Language Technologies Research Center, International Institute of Information Technology, Hyderabad, India 500032.

Brants, TnT – ***A statistical part-of-speech tagger***. In Proc. Of the 6th Applied NLP Conference, pp. 224-231, 2000.

Cardona, George, 1973, **'Indian grammarians on adverbs'**, *Issues in Linguistics*: papers in honour of Henry and Renee Kahane, ed. by Braj Kachru et. al., Urbana, p. 93.

Chachoo Manzoor Ahmad, S. M. K. Quadri, 2012. **'*Adaptive Hybrid POS Cache based Semantic Language Model'***, International Journal of Computer Applications (0975 – 8887) Volume 39– No.13.

Chandra nitish, Sudhakar kumawat, Vinayak srivastava, ***"Various tagsets for Indian languages and Their performance in Part of speech tagging",*** Department of computer science & engineering, IIIT(BHU), Varanasi

Chirag Patel, Karthik Gali (2008) **"*Part-Of-Speech Tagging for Gujarati Using Conditional Random Fields*".** In Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages, pages 117–122,

Cloeren, Jon, 1999, 'Tagsets' in 'Syntactic Word-class Tagging', Kluwer Academic Publishers, Dordrecht, The Netherlands, p.38-39

Das, B. R., & Patnaik, S. (2014*),* ***"A Novel Approach for Odia Part of Speech Tagging Using Artificial Neural Network".*** In Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2013, pp. 147-154. Springer International Publishing

Dash, Sinuruddha, (1995), The Syntax and Semantics of Sanskrit Nominal Compounds, University of Madras, Madras, p.30-31

Debasri Chakrabarti (2011), ***Layered Parts of Speech Tagging for Bangla***, Language in India www.languageinindia.c o m, M a y 2 0 1 1, Special Volume:Problems of Parsing in Indian Languages.

Delip Rao and David Yarowsky (2007), ***Part of Speech Tagging and Shallow Parsing of Indian Languages***, Department of Computer Science, Johns Hopkins University, USA, 2007.Proceedings of workshop on Shallow Parsing in South Asian Languages shiva .iiit.ac.in/SPSAL2007/final/iitmcsa.pdf.

Dermatas and K. George, ***Automatic stochastic tagging of natural language texts***. Computational Linguistics, 21(2): 137-163, 1995.

Dhanalakshmi V, Anand Kumar, Shivapratap G, Soman KP and Rajendran S (2009***), Tamil POS Tagging using Linear Programming***, International Journal of Recent Trends in Engineering, Vol. 1, No. 2, May 2009.

Dinesh Kumar and Gurpreet Singh Josan, (2010), ***Part of Speech Taggers for Morphologically Rich Indian Languages: A Survey,*** International Journal of Computer

Applications (0975 – 8887) Volume6–No.5, September, 2010, www.ijcaonline.org/ volume6/number5 /pxc3871409 .pdf.

Dutta Pallav Kumar,(2013) ***"An online Semi Automated Part of Speech Tagging Technique applied To Assamese",*** Thesis submitted to IIT Guwahati India.

E. Dermatas and K. George, ***Automatic stochastic tagging of Natural language texts,*** Computational Linguistics, 21(2): 137-163, 1995

Ekbal Asif, et.al***, Bengali Part of Speech Tagging using Conditional Random Field*** in Proceedings of the 7th International Symposium of Natural Language Processing (SNLP-2007), Pattaya, Thailand, 13-15 December 2007, pp.131-136

----------- Bandyopadhyay, S., ***Part of Speech Tagging in Bengali Using Support Vector Machine,*** ICIT- 08, IEEE International Conference on Information Technology, pp. 106-111, 2008

-----------S. Bandyopadhyay***, Lexicon Development and POS tagging using a Tagged Bengali News Corpus***, In *Proc. of FLAIRS-2007*, Florida, 261-263, 2007

G.M. Ravi Sastry , Sourish Chaudhuri and P. Nagender Reddy, ***An HMM based Part-Of-Speech tagger and statistical chunker for 3 Indian languages,*** www.cs.cmu.edu/~schaudhu/publications.html.

Ganesan, M. (1994). **"*Functions of Morphological Analyzer Developed at CIIL, Mysore*"**, in Harikumar Basi (ed.) Automatic Translation (seminar proceedings), Thiruvanthapuram: ISDL.

Garg Navneet, Vishal Goyal, Suman Preet, (2012). ***'Rule Based Hindi Part of Speech Tagger',*** Proceedings of COLING. pages 163–174, COLING 2012, Mumbai.

Girish Nath Jha, (2010) ***"The TDIL program and the Indian Language Corpora Initiative (ILCI)".*** language resource and evaluation conference.

Gopal Madhav, Diwakar Mishra, and Devi Priyanka **Singh *"Evaluating Tagsets for Sanskrit"***, Lecture Notes in Computer Science, 2010

Gopal, Madhav and Jha, Girish N. (2011), ***Tagging Sanskrit Corpus Using BIS POS Tagset.*** In: Proceedings of the International Conference, ICISIL 2011, Patiala, India, March 9-11,2011, CCIS 139 pp 191-194, Heidelberg: Springer.

Gurpreet Singh, ***Development of Punjabi Grammar Checker***, Phd. Dissertation, 2008

Hammad Ali (2010***), An Unsupervised Parts-of-Speech Tagger for the Bangla language***, Department of Computer Science, University of British Columbia. 2010.

------------Haque, R. and S. Bandyopadhyay, ***Named Entity Recognition in Bengali: A Conditional Random Field Approach***, In *Proc. of 3rd IJCNLP*, 51-55, 2008

Hasan, Fahim Muhammad, Naushad UzZaman and Mumit Khan,(2007**). '*Comparison of different POS Tagging Techniques (n-Gram, HMM and Brill's tagger) for Bangla',*** Center for Research on Bangla Language Processing, BRAC University, Bangladesh

Jabar Hassan Yousif , Tengku Mohd Tengku Sembok, ***Arabic part-of-speech tagger based support vectors machines.***

James Allen, ***Natural Language Understanding***, Benjamin/ Cummings Publishing Company, 1995

Javed Ahmed MAHAR Ghulam Qadir MEMON,(2010**)*"Rule Based Part of Speech Tagging of Sindhi Language "*** proceeding of International Conference on Signal Acquisition and Processing.

Javier M. Moguerza and Alberto Munoz, (2006) ***"Support Vector Machines with Applications",*** Statistical Science 2006, Vol. 21, No. 3, 322–336 Institute of Mathematical Statistics

Jes´us Gim´enez and Llu´ıs M`arquez**., *SVMTtool:Technical manual*** *v1.3*, August 2006

Jha Girish Nath '**The System of Panini**' Language in India, volume 4:2 February 2004

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. ***Conditional random fields: Probabilistic models for segmenting and labeling sequence data***. In *Proceedings of the 18th International Conf. on Machine Learning*, pages 282–289.Morgan Kaufmann, San Francisco, CA

Jorg Asmussen, ***Survey of POS taggers*** DK-CLARINWP 2.1 Technical Report,2011

Joshi, S.D., 1966, ***'Adjectives and Substantives as a Single Class in the Part-of-Speech'***, Journal of the University of Poona, Vol.25, pp. 25

Jurafsky D and Marting J H, ***Speech and Language Processing An Introduction to Natural Language Processing,*** Computational Linguistics and Speech Recognition, Pearson Education Series 2002

Kudo, T and Matsumoto, ***Chunking with Support Vector Machines,*** In Proc. of NAACL, 192-199, 2001.

Kumar Ritesh, Lahiri Bornini, Deepak Alok,, ***" Developing a POS tagger for Magahi : A comprehensive study".*** 2014.

Lafferty, J., McCallum, A., and Pereira, F., ***Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,*** In Proc. of the 18th ICML"01, 282-289, 2001.

Leech, Geoffrey and Smith, Nicholas, 1999, ***'The use of Tagging' in 'Syntactic Word-class Tagging'***, Kluwer Academic Publishers, Dordrecht, The Netherlands, p.24

Linda Van Guilder (1995) ***Automated Part of Speech Tagging: A Brief Overview Handout for LING361***, Fall 1995 Georgetown University

M. Selvam, A.M. Natarajan (2009***), Improvement of Rule Based Morphological Analysis and POS Tagging in Tamil Language via Projection and Induction Techniques,*** International Journal of Computers, Issue 4, Volume 3, 2009.

Manish Shrivastava and Pushpak Bhattacharyya, ***Hindi POS Tagger Using Naive Stemming: Harnessing Morphological Information Without Extensive Linguistic Knowledge***, International Conference on NLP (ICON08), Pune, India, December, 2008 Also accessible from http://ltrc.iiit.ac.in/proceedings/ICON-2008

Manjit Kaur, Mehak Aggerwal, and Sanjeev Kumar Sharma, (2015***)," Improving Punjabi Part of Speech Tagger by Using Reduced Tag Set*****",**International Journal of Computer Applications & Information Technology Vol. 7, Issue II Dec 14- January 2015 (ISSN: 2278-7720)

Manju K., Soumya S., Sumam Mary Idicula, ***Development of a POS Tagger for Malayalam - An Experience,*** artcom, pp.709-713, 2009 International Conference on Advances in Recent Technologies in Communication and Computing, 2009

Mona Parakh, Rajesha N. and Ramya M (2011), ***Sentence Boundary Disambiguation in Kannada Texts***, Language in India www.languageinindia.c o m 1 1 : 5 M a y 2 0 1 1 Special Volume:Problems of Parsing in Indian Languages, Pages 17-19.

Namrata Tapaswi Suresh Jain , ***“Treebank Based Deep Grammar Acquisition and Part-Of-Speech Tagging for Sanskrit Sentences”*** Software Engineering (CONSEG), 2012 CSI Sixth International Conference on.

Neetu Aggarwal, Amandeep kaur Randhawa. (2015). ***‘A Survey on Parts of Speech Tagging for Indian Languages’***, International Journal of Computer Applications, International Conference on Advancements in Engineering and Technology.

----------, K.P Soman,(2011) **Part of Speech Tagging For Indian Language :A Literary Survey,**www.research.ijcaonline.org/volume34/number8/pxc3875993.pdf

Nidhi Mishra Amit Mishra (2011***), Part of Speech Tagging for Hindi Corpus,*** International Conference on Communication Systems and Network Technologies.

Oliver Hellwig: ***Performance of a lexical and POS tagger for Sanskrit.*** In: *Proceedings of the Fourth International Sanskrit Computational Linguistics Symposium*, pp. 162-172.

Oliver Hellwig: ***SanskritTagger, a stochastic lexical and POS tagger for Sanskrit.*** In: *Proceedings of the First International Sanskrit Computational Linguistics Symposium*, pp. 37-46

Pallavi Bagul, Archana Mishra, Prachi Mahajan, Medinee Kulkarni, Gauri Dhopavkar, ***"Rule Based POS Tagger for Marathi Text"*** in proceeding of: International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 1322-1326.

Pattabhi R K Rao T, Vijay Sundar Ram R, Vijayakrishna R and Sobha L (2007), ***A Text Chunker and Hybrid POS Tagger for Indian Languages,*** AU-KBC Research Centre, MIT Campus, Anna University, Chromepet, Chennai, 2007 . shiva.iiit.ac.in/SPSAL2007/final/aukbc.pdf.

Pradipta Ranjan Ray, Harish V., Sudeshna Sarkar and Anupam Basu, "***Part of Speech Tagging and Local Word Grouping Techniques for Natural Language Parsing in Hindi",*** Department of Computer Science & Engineering, Indian Institute of Technology, Kharagpur, INDIA 721302. www.mla.iitkgp.ernet.in/papers/hindipostagging.pdf

PVS Avinesh, G Karthik, ***Part-Of-Speech Tagging and Chunking using Conditional Random Fields and Transformation Based Learning*** in the proceedings of NLPAI Contest, 2006

Rajendran S. (2006), ***Parsing in Tamil***, LANGUAGE IN INDIA www.languageinindia.com Volume 6: 8 August, 2006.

RamaSree, R.J, Kusuma Kumari, P., ***Combining Pos Taggers For Improved Accuracy To Create Telugu Annotated Texts For Information Retrieval***, 2007, Available at http://www.ulib.org/conference/2007/RamaSree.pdf

Rathod Shubhangi, Sharvari Govilkar, (2015), ***"Survey of various POS tagging techniques for Indian regional languages"***, (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (3) , , 2525-2529

Ratnaparkhi, A., ***A Maximum Entropy Part of Speech Tagger***, In Proc. of the EMNLP Conference, 133-142, 1996

Rose Hu Xunlei, Eric Atwell, (2003). ***'A survey of machine learning approaches to analysis of large corpora',*** School of Computing, University of Leeds, U.K. LS2 9JT

Roy Bipul, Bipul Syam Purkayastha***, A Study on Different Part of Speech (POS) Tagging Approaches in Assamese Language,*** International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 3, March 2016

S. Rajendran (2006), ***"Parsing in Tamil***", LANGUAGE IN INDIA ww.languageinindia.com Volume 6: 8 August, 2006

S. Singh , K. Gupta , M. Shrivastava and P. Bhattacharya, ***Morphological Richness Offsets Resource Demand- Experiences in Constructing a POS Tagger for Hindi***, In Proc. of COLING/ACL, 779-786, 2006

Sandipan Dandapat, Sudeshna Sarkar, Anupam Basu***, Automatic Part-of-Speech Tagging for Bengali: An Approach for Morphologically Rich Languages in a Poor Resource Scenario,*** Proceedings of the Association for Computational Linguistic, pp 221-224, 2007

Sathish Chandra Pammi and Kishore Prahallad (2007), ***POS Tagging and Chunking using Decision Forests***, Workshop on shallow parsing in South Asian languages, 2007. shiva.iiit.ac.in/SPSAL2007/proceedings.php.

Shambhavi B R, RamakanthKumar 2012. ***"Kannada Part-Of-Speech Tagging with Probabilistic Classifiers"***. International Journal of Computer Applications (0975 – 888) Volume 48– No.17, June 2012

Sharma S.K, Lehal G.S (2011) **"*Using HMM to Improve accuracy of Punjabi POS tagger*"** 2011 IEEE International Conference on computer science and Automation Engineering. Shanghai (China)

Singh Jyoti, Nisheeth Joshi Iti Mathur, 2013**," Development of Marathi Part of Speech Tagger Using Statistical Approach"**, Advances in Computing, Communications and Informatics (ICACCI).

---------- Nisheeth Joshi Iti Mathur, (2013) ***"Part of speech tagging of marathi text using trigram method"*** International Journal of Advanced Information Technology (IJAIT) Vol. 3, No.2.

Singh Mandeep, Lehal Gurpreet, and Sharma Shiv, 2008. ***A Part-of-Speech Tagset for Grammar Checking of Punjabi,*** published in The Linguistic Journal, Vol 4, Issue 1, pp 6-22

Singha, Kh Raju, Ksh Krishna Bati Singha, Bipul Syam Purkayastha,(2013**), "*Developing a Part of Speech Tagger for Manipuri"***, International Journal of Computational Linguistics and Natural Language Processing, Vol 2 Issue 9 September 2013 ISSN 2279 – 0756

Siva Reddy, Serge Sharoff. (2011) ***"Cross Language POS Taggers (and other Tools) for Indian Languages: An Experiment with Kannada using Telugu Resources".*** In Proceedings of IJCNLP workshop on Cross Lingual Information Access: Computational Linguistics and the Information Need of Multilingual Societies. Thailand, 2011

Smriti Singh, et.al***, Morphological Richness Offsets Resource Demand- Experiences in Constructing a POS Tagger for Hindi***, in the proceedings of COLING/ACL, pp. 779-786, 2006

Sumam Mary Idicula and Peter S David, ***A Morphological processor for Malayalam Language***, South Asia Research, SAGE Publications, 2007

Vijayalaxmi .F. Patil (2010), ***Designing POS Tagset for Kannada, Linguistic Data Consortium for Indian Languages (LDC-IL)***, Organized by Central Institute of Indian Languages, Department of Higher Education Ministry of Human Resource Development, Government of India, March 2010.

Yi lin, 1999, ***"Support vector machine and the bayes rule in the classification",*** technical report no.1014, Department of Statistics University of Wisconsin.

Books

Aiyar, Ramachandra T.K, 2004, ***Bālarāmāyaṇam***, R.S.Vadhyar & Sons, Palghat, South India.

Alpaydin Ethem *(2004(*a*))**Introduction to Machine Learning,*** Prentice-Hall of India Private Limited, New Delhi.

Apte, Vaman Shivaram, 2002, ***The Student's Guide to Sanskrit Composition***, Chowkhaamba Sanskrit Series Office, Varanasi, India

Bharati, Akshar and Vineet Chaitanya, Rajeev Sangal (2004), ***Natural language: Processing:Paninian Perspective,*** Prentice Hall of India Private Limited, New Delhi

Cardona, George, 1997, ***Pāṇini: his work and its traditions***, Motilalal Banarasidass, New Delhi.

Christopher D. Manning and Hinrich Schiitze (1999), ***"Foundation of statistical Natural Language Processing",*** The MIT Press, Cambridge, Massachusetts, London, England

Cloeren, Jon, 1999, 'Tagsets' in ***Syntactic Word-class Tagging***, Kluwer Academic Publishers, Dordrecht, The Netherlands, pp.37-54.

Dash, Sinuruddha, 1995, ***The Syntax and Semantics of Sanskrit Nominal Compounds***, University of Madras, Madras.

George Cardona and Dhanesh Jain (eds.), "***The Indo-Aryan languages"***, Routledge Language Family Series, vol. 2, London and New York: Routledge.

Halteren, Hans van and Voutilainen, Atro, 1999, ***'Automatic taggers: An Introduction***', in ***Syntactic Wordclass Tagging***, Halteren, Hans van (Ed.), Kluwer Academic Publishers, Netherlands, pp.109-115.

Iyer, K.A. Subramania, 1971, "***The Vakyapadiya of Bhartrhari"***, Ch. III, pt. I, English Translation, Deccan College, Pune.

Jain, D. & Cardona, G. (2007). ***"The Indo-Aryan Languages".*** Taylor & Francis.

Jurafsky, Daniel & James H. Martin, 2002, ***Speech and Language Processing***, Pearson Education, Delhi, p.318

Kṛṣṇaśarmā, 1904, "***Vaiyākaraṇa-Siddhānta-Kaumudī***", Nirṇayasāgara Publication, Mumbai.

Kale, M.R, 1995, "***A Higher Sanskrit Grammar***", MLBD Publishers, New Delhi, pp.194-216.

Kale, M.R., 1996, "***Mālatīmādhava of Bhavabhūti***", Motilal Banarsidass, Delhi

Kale, M.R., 1997, "***Daśakumāracarita of Daṇḍin***", Motilal Banarsidass Publishers, Delhi.

Kale, M.R., 1999, "***Pañcatantra of Viṣṇuśarman***", Motilal Banarsidass Publishers Pvt. Ltd., Delhi

Kannan, K.S., 1983, ***Samskrita Dhaturupakosha – Savivarana***, Bharati Prakashana, Bangalore 41, p.61

Mallikarjun, B and Mohamed Yoonus M, Samar Sinha, Vadivel A (2010), ***Indian Languages and Part-of Speech Annotation,*** Central Institute of Indian Languages, Mysore

Patterson, Dan W (2005), ***Introduction to Artificial Intelligence and Expert Systems,*** Prentice Hall of India Private Limited, New Delhi

Paul Kiparsky, ***Paninian Linguistics***. *Encyclopedia of Languages and Linguistics*, 1993

Sharma, Rama Nath, 2003, ***The Aṣṭādhyāyi of Pāṇini***, (6 Vols.) Munshiram Manoharlal Publishers Pvt. Ltd., Delhi.

Sharma, Rama Nath, ***The Aṣṭādhyāyī of Pāṇini*** – Volume-I page-75-76

Spiejer, J.S., 1886, ***Sanskrit Syntax***, Reprint 1998, Motilal Banarsidas, Delhi – 07

Voutilainen, Atro, 1999, ***'A Short History of Tagging'*** in Syntactic Wordclass Tagging, Ed. Halteren, Hans van, Kluwer Academic Publishers, Netherlands p.10

Theses and Dissertations:

देवि प्रियंका सिंह(2011), ***हिन्दी वर्ग-समूहों का तुलनात्मक अध्ययन,*** M.Phil Dissertation, JNU, New Delhi

Andrew Hardie (2003),***The computational analysis of morphosyntactic categories in Urdu,*** Ph.D Thesis , Lancaster University

Behera, P. (2015***). "Odia Parts of Speech Tagging Corpus: Suitability of Statistical Models***. *M. Phil. Dissertation"*. New Delhi: Jawaharlal Nehru University.

DeRose, S. J. (1990). "***Stochastic Methods for Resolution of Grammatical Category Ambiguity in Inflected and Unified Languages".*** Ph.D. Dissertation. Department of Cognitive and Linguistic Sciences, Providence, Brown University.

DeRose, S. J. (1990). "***Stochastic Methods for Resolution of Grammatical Category Ambiguity in Inflected and Unified Languages"***. Ph.D.Dissertation. Department of Cognitive and Linguistic Sciences, Providence, Brown University.

MacKinlay Andrew, ***"The Effects of Part-of-Speech Tagsets on Tagger Performance",*** thesis Submitted to The Department of Computer Science and Software Engineering ,University of Melbourne, 2005

R. Chandrashekar, (2007), "***Part-of-Speech Tagging for Sanskrit,"*** Ph.D. Thesis, JNU, New Delhi

Sachin Kumar(2012), *"**Named Entities Recognition for Sanskrit: A Hybrid Approach",*** Ph.D Thesis, JNU, New Delhi

Singh, D. P. (2011). "***A Comparative Study of Hindi Parts of Speech Tagsets"***. M.Phil Dissertation, Centre for Linguistics, J.N.U., New Delhi.

Singh, Srishti. (2015). ***"Challenges in Automatic POS Tagging of Indian Languages- A Comparative Study of Hindi and Bhojpuri".*** M. Phil. Dissertation. New Delhi: Jawaharlal Nehru University.

Subash (2006), ***Machine Recognition and Morphological Analysis of Subanta padas,*** M.Phil Dissertation, JNU, New Delhi

Online Source

1. http://bllip.cs.brown.edu/papers/charniak97statistical.pdf
2. http://bowlandiles.lancs.ac.uk/monkey/ihe/linguistics/corpus2/2fra1.htm
3. http://en.wikipedia.org/wiki/Brill_tagger
4. http://en.wikipedia.org/wiki/Malayalam
5. http://en.wikipedia.org/wiki/Pos-tagging
6. http://en.wikipedia.org/wiki/Punjabi_grammar
7. http://en.wikipedia.org/wiki/Punjabi_language
8. http://en.wikipedia.org/wiki/Telugu_language
9. http://en.wikipedia.org/wiki/Telugu_language
10. http://ilk.uvt.nl/mbt/
11. http://language.worldofcomputing.net/pos-tagging/markov-models.html
12. http://language.worldofcomputing.net/pos-tagging/parts-of-speech-tagging.html
13. http://nlp.stanford.edu/software/tagger.shtml
14. http://nptel.iitm.ac.in/courses/106101007/2
15. http://research.ijcaonline.org/volume34/number8/pxc3875993.pdf
16. http://sanskrit.jnu.ac.in/post/post.jsp
17. http://sanskritessays.blogspot.in/
18. http://sudharma.epapertoday.com/
19. http://tomcat.apache.org/
20. http://ucrel.lancs.ac.uk/

21. http://ucrel.lancs.ac.uk/claws/

22. http://uima.apache.org/sandbox.html

23. http://wortschatz.uni-leipzig.de/~cbiemann/software/unsupos.html

24. http://wortschatz.uni-leipzig.de/~cbiemann/software/unsupos.html

25. http://www.aaai.org/AITopics/html/machine.html

26. http://www.bangla-online.info/PromotionalSite/Bangla

27. http://www.baraha.com/BarahaIME.htm

28. http://www.comp.lancs.ac.uk/ucrel/claws/trial.html

29. http://www.comp.lancs.ac.uk/ucrel/claws/trial.html

30. http://www.indsenz.com/int/index.php?content=sanskrit_tagger

31. http://www.languageinindia.com/feb2004/panini.html

32. http://www.ldc.upenn.edu/annotation’

33. http://www.lsi.upc.edu/~nlp/SVMTool/

34. http://www-nlp.stanford.edu/links/statnlp.html

35. https://balramshukla.wordpress.com/

36. https://sites.google.com/site/praachiprajnaa/home

37. Language/IntroductionOfBanglaLanguage.htm

APPENDICES

Appendix I. Representative Set of SVM Data Used for Training

No.	**Training Data**	**Tokenized Data**
1.	तथापि RP_RPD	ततः
2.	मे PR_PRL	तस्य
3.	दिल्याम् N_NNP	जातायां
4.	अन्यूनः QT_QTF	सहसा
5.	प्रणयः N_NN	स्वयम्
6.	अतो RP_RPD	।
7.	रक्षेम V_VM_VF	भगवन्तं
8.	दिल्लीं N_NNP	प्रणम्य
9.	नूतन JJ	अथ
10.	रुधिर N_NN	विस्मितः
11.	समर्पणेन N_NN	प्रययौ
12.	अवरोधान् N_NN	नृपः
13.	अरिहृत्य V_VM_VNG	।
14.	आनन्दान्वितरेम V_VM_VF	तीव्र
15.	। RD_PUNC	व्रते
16.	मार्गयो: N_NN	अपि
17.	उभयोः DM_DMD	श्रोणस्य
18.	प्रवृध्दैः JJ	कदाचित
19.	वृक्षै: N_NN	समजायत
20.	संलापः N_NN	।
21.	कुर्म: V_VM_VF	स
22.	। RD_PUNC	तम
23.	भारतराष्ट्रस्य N_NNP	आहूय
24.	विनूत्नद्वाररूपेण N_NNP	भगवान्
25.	निर्मीमः V_VM_VF	विलक्षं
26.	दिल्लीम् N_NNP	प्राह
27.	। RD_PUNC	सस्मितः
28.	जनगणमनोभिलाषाः N_NNP	।
29.	डा.सुरेन्द्रमोहनमिश्रः N_NNP	को
30.	संस्कृतभारति N_NNP	अयं
31.	! RD_SYM	परिवितर्क:
32.	संस्कृतभारति N_NNP	ते

33.	! RD_SYM	प्रतिसंलीन
34.	संस्कुरु V_VM_VF	चेतसः
35.	विश्वमजस्रम् N_NN	।
36.	। RD_PUNC	विश्लिष्ट
37.	संहति N_NN	अत्यन्तकृष्टा
38.	संस्कृति N_NN	वा
39.	धर्मसदर्थान् N_NN	तन्त्री
40.	जनगणमनोभिलाषान् N_NNP	बहवति
4.	॥ RD_PUNC	विस्वरा
42.	तिमिरम् N_NN	।
43.	अपाकुरु V_VM_VF	समा
44.	रजनीं N_NN	माधुर्यम्
45.	राकाभासुरभविकाम् N_NNP	आयाति
46.	॥ RD_PUNC	तस्मात्
47.	तस्य PR_PRP	साम्यं
48.	विजयस्य N_NN	सम
49.	स्मरणाय N_NN	आश्रयेत्
50.	प्रतिवर्षम् N_NN	।
51.	इयं DM_DMD	इति
52.	दुर्गापूजा N_NN	आदेशाद्
53.	अनुष्ठिता JJ	भगवतः
54.	भवति V_VM_VF	सर्वसाम्यम्
55.	| RD_PUNC	उपागतः
56.	अनेकेषु QT_QTF	।
57.	प्रतिवर्षम् RP_RPD	स
58.	अस्मिन्नवसरे N_NN	प्राप
59.	रामलीलायाः N_NN	विमलज्ञानं
60.	प्रदर्शनं N_NN	पश्चात्ताप
61.	" RD_SYM	विवर्जितः
62.	विजयादशमी N_NNP	।
63.	कथ्यते V_VM_VF	तस्य
64.	| RD_PUNC	ताम्
65.	वस्त्राणि N_NN	अद्भुतां
66.	धारयन्ति V_VM_VF	सिद्धिं
67.	| RD_PUNC	विलोक्य

68.	विविधानि RP_RPD	पृथुविस्मयाः
69.	मधुराणि N_NN	।
70.	खादन्ति V_VM_VF	पप्रच्छुः
71.	खादयन्ति V_VM_VF	भिक्षवः
72.	च CC_CCD	सर्वे
73.	| RD_PUNC	भगवन्तंस
74.	दशम्यामेव N_NNP	च
75.	दुर्गायाः N_NNP	अभ्यधात्
76.	प्रतिमाः N_NN	।
77.	जले N_NN	श्रोणस्य
78.	विसर्जिताः V_VM	श्रूयताम्
79.	भवन्ति V_ V_VAUX	श्रेयः
80.	| RD_PUNC	कर्म
81.	गृहे N_NN	जन्मान्तर
82.	वेदानां N_NN	अर्हितम्
83.	च CC_CCD	।
84.	रामायणस्य N_NNP	न
85.	च CC_CCD	ह्यपुण्य
86.	पाठं N_NN	अनुभावानां
87.	कुर्वान्ति V_VM	भवन्ति
89.	स्म V_VAUX	अद्भुत
90.	आबल N_NN	संपदः
91.	वृद्धाः N_NN	।
92.	तस्मात् DM_DMD	विपश्वी
93.	वेदाः N_NN	भगवान्
94.	भारतीय N_NNP	सम्यक्
95.	संस्कृतिः N_NN	संबुद्धः
96.	तथा CC_CCD	सुगतः
97.	आधारशिलाः JJ	पुरा
98.	वर्तन्ते V_VM_VF	।
99.	उपनिषद N_NN	पुरीं
100.	गीता N_NNP	बन्धुमतीं

Appendix II

Representative Set of SVM Data Used Testing and Evaluation

NO.	GOLD DATA	Tagger Output
1.	भक्त्या N_NN	भक्त्या N_NN
2.	तथागतं N_NN	तथागतं N_NN
3.	द्रष्टुं V_VM_VINF	द्रष्टुं N_NN
4.	पद्भ्यामेव N_NN	पद्भ्यामेव N_NN
5.	महीपतिः N_NN	महीपतिः N_NN
6.	। RD_PUNC	। RD_PUNC
7.	अस्पृष्टपादस्य N_NN	अस्पृष्टपादस्य N_NN
8.	भुवा N_NN	भुवा V_VM_VNG
9.	श्रुणस्य N_NN	श्रुणस्य N_NN
10.	आजन्मवासरात् N_NN	आजन्मवासरात् V_VM_VF
11.	। RD_PUNC	। RD_PUNC
12.	महार्हवस्त्रैः N_NN	महार्हवस्त्रैः N_NN
13.	प्रस्थाने N_NN	प्रस्थाने N_NN
14.	भृत्यैः N_NN	भृत्यैः N_NN
15.	आच्छादिता V_VM_VF	आच्छादिता N_NN
16.	मही N_NN	मही N_NN
17.	। RD_PUNC	। RD_PUNC
18.	भगवद्भक्ति N_NN	भगवद्भक्ति V_VM_VF
19.	विनयाद् N_NN	विनयाद् N_NN
20.	गौरवाच्च N_NN	गौरवाच्च N_NN
21.	स PR_PRP	स PR_PRP
22.	भूपतेः N_NN	भूपतेः N_NN
23.	। RD_PUNC	। RD_PUNC
24.	वस्त्राणि N_NN	वस्त्राणि N_NN
25.	अवारयद् V_VM_VF	अवारयद् N_NN
26.	भृत्यैरवाच्च N_NN	भृत्यैरवाच्च N_NN
27.	इव RP_RPD	इव RP_RPD
28.	क्षितौ N_NN	क्षितौ N_NN
29.	। RD_PUNC	। RD_PUNC
30.	वारितेष्वथ N_NN	वारितेष्वथ N_NN
31.	वस्त्रेषु N_NN	वस्त्रेषु N_NN

32.	दिव्यवस्त्रैः N_NN	दिव्यवस्त्रैः JJ
33.	वृता V_VM_VF	वृता N_NN
34.	मही N_NN	मही JJ
35.	। RD_PUNC	। RD_PUNC
36.	अप्रयत्नोपकरणाः JJ	अप्रयत्नोपकरणाः JJ
37.	संपदः N_NN	संपदः N_NN
38.	पुण्यशालिनाम् N_NN	पुण्यशालिनाम् N_NN
39.	। RD_PUNC	। RD_PUNC
40.	निवार्य V_VM_VNG	निवार्य N_NN
4.	दिन्यवस्त्राणि N_NN	दिन्यवस्त्राणि N_NN
42.	भूमौ N_NN	भूमौ N_NN
43.	तेन PR_PRP	तेन PR_PRP
44.	अर्पिते V_VM_VF	अर्पिते V_VM_VF
45.	पदे N_NN	पदे N_NN
46.	। RD_PUNC	। RD_PUNC
47.	विचचालाचला N_NN	विचचालाचला N_NN
48.	पृथ्वी N_NN	पृथ्वी N_NN
49.	सशैलवनसागरा N_NN	सशैलवनसागरा N_NN
50.	। RD_PUNC	। RD_PUNC
51.	ततः RP_RPD	ततः N_NST
52.	स PR_PRP	स PR_PRP
53.	भूमिपतिना N_NN	भूमिपतिना N_NN
54.	सह PSP	सह PSP
55.	प्राप्य V_VM_VNG	प्राप्य V_VM_VNG
56.	जिन N_NN	जिन N_NN
57.	अश्रमम् V_VM_VF	अश्रमम् V_VM_VF
58.	। RD_PUNC	। RD_PUNC
59.	भगवन्तं N_NN	भगवन्तं N_NN
60.	विलोक्य V_VM_VNG	विलोक्य N_NN
61.	अस्य N_NN	अस्य PR_PRP
62.	विदधे V_VM_VF	विदधे N_NN
63.	पादवन्दनम् N_NNV	पादवन्दनम् N_NNV
64.	। RD_PUNC	। RD_PUNC
65.	उपविष्टस्य JJ	उपविष्टस्य N_NN
66.	तस्य PR_PRP	तस्य PR_PRP

67.	अग्रे N_NST	अग्रे PSP
68.	हृष्टस्यालोकनामृतैः N_NN	हृष्टस्यालोकनामृतैः N_NN
69.	। RD_PUNC	। RD_PUNC
70.	चक्रे N_NN	चक्रे N_NN
71.	शमविवेकस्य N_NN	शमविवेकस्य N_NN
72.	भगवान N_NN	भगवान N_NN
73.	अभिषेचनम् N_NNV	अभिषेचनम् N_NNV
74.	। RD_PUNC	। RD_PUNC
75.	आशयानुशयं N_NN	आशयानुशयं N_NN
76.	धातुं N_NN	धातुं N_NN
77.	प्रकृतिं N_NN	प्रकृतिं N_NN
78.	च CC_CCD	च CC_CCD
79.	विचार्य V_VM_VNG	विचार्य V_VM_VNG
80.	सः PR_PRP	सः PR_PRP
81.	। RD_PUNC	। RD_PUNC
82.	सत्यसंदर्शनाय N_NN	सत्यसंदर्शनाय N_NN
83.	अस्य PR_PRP	अस्य PR_PRP
84.	विदशे N_NN	विदशे N_NN
85.	धर्मदेशनाम् N_NN	धर्मदेशनाम् N_NN
86.	। RD_PUNC	। RD_PUNC
87.	सत्काय N_NN	सत्काय N_NN
89.	दृष्टिशैलो N_NN	दृष्टिशैलो N_NN
90.	अस्य PR_PRP	अस्य PR_PRP
91.	तया PR_PRP	तया PR_PRP
92.	विंशति QT_QTC	विंशति V_VM_VF
93.	शृङ्गवान् V_VM_VF	शृङ्गवान् N_NN
94.	। RD_PUNC	। RD_PUNC
95.	ज्ञानवज्रेण N_NN	ज्ञानवज्रेण N_NN
96.	निर्भिन्नः N_NN	निर्भिन्नः N_NN
97.	स्रोतः N_NN	स्रोतः N_NN
98.	प्राप्ति N_NN	प्राप्ति V_VM_VF
99.	पदस्पृशः N_NN	पदस्पृशः N_NN
100.	। RD_PUNC	। RD_PUNC

Appendix III

BUREAU OF INDIAN STANDARDS GUIDLINE FOR SASNKRIT WITH EXAMPLE

Sl. No	**Category**			**Label**	**Annotation Convention****	**Remarks**
	Top level	Subtype (level 1)	Subtype (level 2)			
1	**Noun**			**N**	**N**	
1.1		Common		NN	N__NN	
1.2		Proper		NNP	N__NNP	Tasya/PR_PRL **Somadattā/N_NNP** nāma/N_NN bhāryā /N_NN āsīt/V_VM_VF |/RD_PUNCH
1.3		Verbal		NNV	N_NNV	The verbal noun ub type is only for languages such as Tamil and Malyalam)
1.4		Nloc		NST	N__NST	
2	**Pronoun**			**PR**	**PR**	
2.1		Personal		PRP	PR__PRP	
2.2		Reflexive		PRF	PR__PRF	
2.3		Relative		PRL	PR__PRL	
2.4		Reciprocal		PRC	PR__PRC	
2.5		Wh-word		PRQ	PR__PRQ	

3	**Demonstrative**			**DM**	**DM**	
3.1		Deictic		DMD	DM__DMD	
3.2		Relative		DMR	DM__DMR	
3.3		Wh-word		DMQ	DM__DMQ	
4	**Verb**			**V**	**V**	
4.1		Main		VM	V__VM	
04/01/01			Finite	VF	V__VM__VF	
04/01/02			Non-finite	VNF	V__VM__VNF	
04/01/03			Infinitive	VINF	V__VM__VINF	
04/01/04			Gerund	VNG	V__VM__VNG	
4.2		Auxiliary		VAUX	V__VAUX	
5	**Adjective**			**JJ**		
6	**Adverb**			**RB**		Only manner adverbs
7	**Postposition**			**PSP**		
8	**Conjunction**			**CC**	**CC**	
8.1		Co-ordinator		CCD	CC__CCD	
8.2		Subordinator		CCS	CC__CCS	
08/02/01			Quotative	UT	CC__CCS__UT	
9	**Particles**			**RP**	**RP**	

9.1		Default		RPD	RP__RPD	
9.2		Classifier		CL	RP__CL	
9.3		Interjection		INJ	RP__INJ	
9.4		Intensifier		INTF	RP__INTF	
9.5		Negation		NEG	RP__NEG	
10	**Quantifiers**			**QT**	**QT**	
10.1		General		QTF	QT__QTF	
10.2		Cardinals		QTC	QT__QTC	
10.3		Ordinals		QTO	QT__QTO	
11	**Residuals**			**RD**	**RD**	
11.1		Foreign word		RDF	RD__RDF	A word written in script other than the script of the original text
11.2		Symbol		SYM	RD__SYM	For symbols such as $, & etc
11.3		Punctuation		PUNC	RD__PUNC	Only for punctuations
11.4		Unknown		UNK	RD__UNK	
11.5		Echowords		ECH	RD__ECH	