Vibhakti Identification Techniques for Sanskrit

Shweta A. Patil
Information Technology Department
Pillai's Institute of Information Technology
New Panvel, Navi Mumbai, India.

VarunakshiBhojane
Computer Engineering Department
Pillai's Institute of Information Technology
New Panvel, Navi Mumbai, India.

***Abstract*-Natural language processingis the branch of computer science focused on developing systems that allow computers to communicate with people using everyday language. Sanskrit is considered as the mother language for almost all Indian languages (indo Aryan). Sanskrit is source ofvast knowledge that knowledge can be accessed by developing computational tools.Sanskrit is grammatically well structured and rich in its inflections.One of the key task in vibhakti identification is identifying the correct root word from its inflected form. In Sanskrit language, these inflected words follow the rules. These rules can be formulated in computational model for developing system to perform vibhakti analysis. This paper gives a survey of various sanskrit vibhakti analysis tools also this paper presents different algorithms and approaches used to develop vibhakti analysis system.**

Keywords*—*Natural language processing, Vibhakti, Sanskrit.

I. INTRODUCTION

Natural language processing, refers to the use and ability of systems to process sentences in a natural language, rather than in a specialized artificial computer language. Many challenges in NLP is enabling computers to derive meaning from human or natural language input. The Sanskrit grammar is the most ancient, strong and simple grammar of the world. Subanta Recognition and Analysis System based on the subanta formulations of Paini. Panini is the author of Sanskrit Grammar, made it, formulating 3,949 rules. Katyakan supplemented the work of Panini with vartka[11].Sanskrit is very rich in its inflections.The inflection involves formation of two kinds of words or padas: subanta padas (nominal words) and tianta padas (verb forms). A word cannot be used in the language unless it is one of them.Vibhakti identification of Sanskritis the basic requirement for the processing of Sanskrit and understanding of Sanskrit. Nominal inflection deals with combination of bases with case suffixes.[4]These extracted suffixes carry sufficient amount of syntactic and semantic information with them.

In vibhakti analysis, system will check punctuations.Then the avyayas and the verbs are recognized by providing databases. After the recognition of these padas, the system recognizes all remaining words as subanta and sends for the analysis process. According to Panini, there are 21 morphological suffixes which are attached to the nominal bases according to syntactic category, gender and end character of the base.

II. BACKGROUD

In the past decade vibhakti analyzers for different languages were developed .Most of the research work has been done on English language computation. Although some research work related to sanskrit language has been done and there is work going on towards developing many computational toolkits and research.

The initial work on sanskrit computation in India was performed at various locations like IIT Kanpur, NCST Mumbai, Computer and Information Science department of Hyderabad, CDAC Pune, Ministry of Communication, IT Government of India. In the mid 90's and late 90's some more projects also started at IIT Bombay, IIT Hyderabad, department of computer science and Engineering Jadavpur University, Kolkata, JNU New Delhi etc.[12]

A. Existing Sysytem

Extracting and annotating vibhakti information is the first step towards understanding the language. Therefore vibhakti identification is the basic tool needed for any NLP applications ranging from information retrieval, search engine, spell checkers to machine translation systems. In this section we cite some existing system and various features of those systems.

1. Subanta Analyzer [1]

Subanta analyzer was designed by Sudhir K Mishra for making a translation tool for Sanskrit language at JNU Delhi[2].The system analyzes inflected noun forms and verb-forms in any given sandhi free text . They have done comprehensive research on the subantarule of Panini and developing the rule base. The system was not able to resolve the ambiguities and also not able to work on Sandhi and Samasa for Sanskrit language .The system has some limitations like. This system does not have gender information for pratipadikas, nor does it attempt to guess the gender. [4]

2. Desika [13]

The Indian Heritage Group of the Centre for Development of Advanced Computing (CDAC) has developed a system called DESIKAisPaninian bases system, which claims to process all the words of Sanskrit and

Includes generation and analysis (parsing)[4].DESIKA includes vedic processing and

TABLE 1.SUMMARY OF SANSKRIT COMPUTATIONAL TOOLS [4,12]

Basis	**Subanta analyzer**	**DESIKA**	**E TRANS**
Features	The system analyzes inflected noun forms and verb forms in any given sandhi free text.	It includes Vedic processing and shabdabodha as well.	The Etrans system comprises of user interface developed.
Approach	Rule Based and example based	rule based approach.	Rule Based approach
Advantages	The average accuracy of the SRAS at this point is 91.65%	Easy implementation and small memory requirement.	Translation of Simple And compound sentences from English to Sanskrit
Limitations	System accepts only non-joint (sandhi-rahita) Sanskrit text. This system fully depends on both the rule base, example base .	Rule Based approach not sufficient due to no morphological order of Sanskrit.	Heavy dependency on Databases.
Future Scope	The average accuracy will improve with several additions to the rule base, example base and other linguistic resource files.	Can be combined with other approaches to make system faster and more robust.	Processengine can be developed with the combination of a stronger rule base and other techniques.

shabda-bodha .In analysis, the syntactic identification and assignment of roles for every word is carried out using the Karaka Vibhakti mappings However, the system (as available at the TDIL site)[12]. This module has two modes: choose mode declines for the pratipadikas already present in the list and edit mode declines for the entries which are not present in the list for which suitable gender and paradigm should be selected. The system has subanta generation only and even that does not work properly.

3. Etrans System [2]

A string of English sentence can be translated into string of Sanskrit sentences. The method for the design and development is in the form of software called "EtranS". The system comprises of user interface developed using .NET framework and the lexicon using MS-Access 2007[12]. The modules are developed on the basis of process engine. The user interface is responsible for taking input and generating the output. It is completely dependent on the tables created in

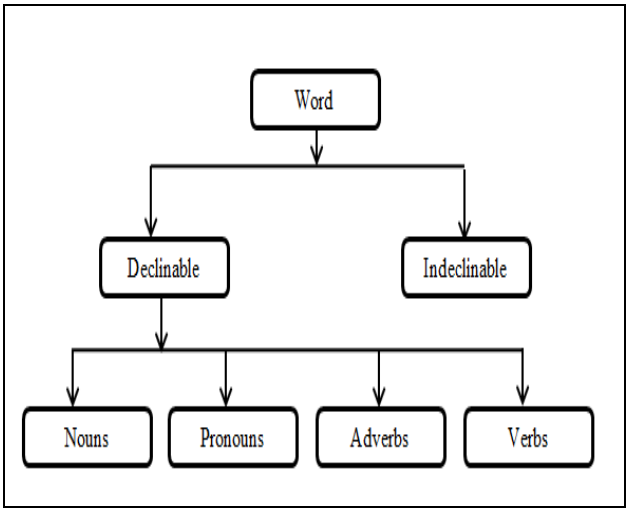

Fig.1.Classification of word [13]

the database for generating output and the programming doneto extract the information based on the logic developed.

III. INTRODUCTION TO SUBANTA

In Sanskrit, the words can be broadly classified into two categories viz. Declinable and Indeclinable. Declinable or inflectional words are those words whose base form can be changed or inflected.[14] For example, the base form or pratipadika can be inflected in 8 vibhaktis. Thesedeclinable/inflectional words can again be categorized as Nouns, Pronouns, Adverbs and Verbs. In Sanskrit, indeclinable wordsare called avyayas. The figure.1 illustrates this. In a Sanskrit sentence, all non-verb categories are subanta-padas. A syntactic unit is called pada. Pada can be nominal (subanta) or verbal (tianta).[1]These forms are formed by inflecting the stems and hence they are part of Sanskrit inflectionalmorphology. According to Paini, there are 21 morphological suffixes (seven vibhaktis and combination of three numbers = 21) which are attached to the nominal bases (pratipadika) according to syntactic category, gender and end character of the base.[4]

Nominal inflection morphology (subanta) deals with combination of bases (pratipadika(with case affixes (sup). The words (padas) thus formed are called subanta.

For example, rama, syama, pustakalaya, vidyalayaetcetc.

Panini has listed the sup suffixes .These suffixes are in the sets of three as- (su, au, jas) (am, au, sas) (a, bhyam, bhis) (e, bhyam, bhyas) (asi, bhyam, bhyas) (as, os, am) (i, os, sup) for singular, dual and plural respectively. These suffixes are added to the pratipadikas (any meaningful form of a word, which is neither a root nor a suffix) to obtain inflected forms (subanta padas).[6]

A. Subanta Recognition Mechanism

First of all, the system checks for punctuations. Then the avyayas and the verbs arerecognized. After the recognition of

these words(padas), the system recognizes all remaining words assubanta and sends for the analysis process.

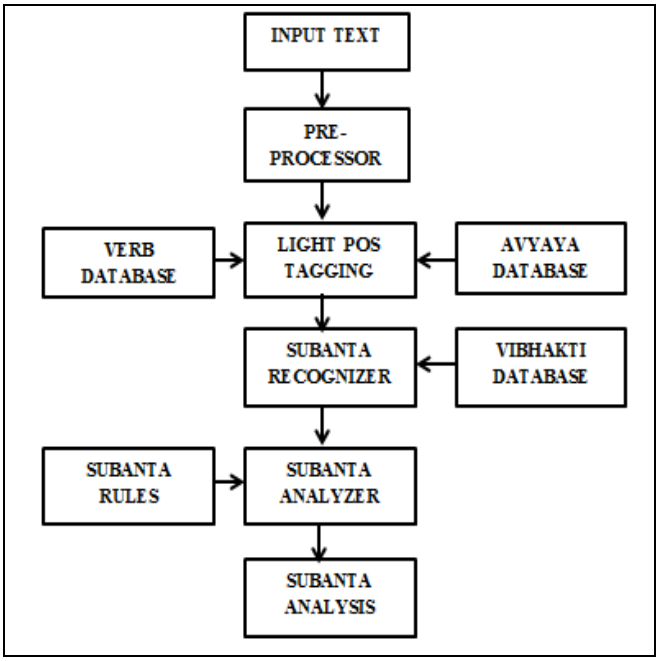

Fig 2 .Subanta recognition process [6]

Input Text:

दैवेनतस्मैसर्वंदत्तम्आसीत्

Output Text:

{ दैवेन] दैव) पुल्लिङ्ग +(टा, तृतीया, एकवचन [तस्मै]तद्+ङेचतुर्थीएकवचन [सर्वं] सर्व+सु, प्रथमाद्वितीयाएकवचन)नपु(., अमद्वितीयाएकवचन) पु [(.दत्तम्] दत्त) पुल्लिङ्ग +(अम्, द्वितीया, एकवचन[[आसीत् VERB] }

- Recognition of punctuations and non-subanta words

System will recognize punctuations and tag them with _PUNCT as a label. If the input has any extraneous characters, then the input word will be cleaned from these elements (normalized) sothat only Devanagari Sanskrit input text is sent to the analyzer[4].

For example

रा&&@?@[[म:, दे*%@व: =राम:,देव:

- Avyaya recognition

If an input word is found in the avyaya database it will marked as Avyaya and not sent to the subantaanalyzer for further processing.

कश्चित्, अथवा, इत्, स्वित्इदानीम्

- Verb recognition

Sanskrit verb forms are very complex they carry tense, aspect, number information all inthe inflection forms. Sanskrit has about 2000 verb roots classified in 10 morphological andsemantic classes. [4]

भवति, अभूवम्,भवतासि, अभूत्

- Subanta recognition

Thus the subanta-padas in Sanskrit text are identified. Further processing is done by comparing words in Rules database and example database.

The Subanta recognition flow shows in fig.2.

IV. VIBHAKTI RULES

Panini is the author of Sanskrit Grammar.It provides a 4000 rules for sanskrit [11]. The rules for vibhakti identification discussed in this section.
Pratipadika (base) suffix structure changes according to the following information

- category of the pratipadika (nouns, pronouns)
- gender of the pratipadika (masculine, feminine, neuter)
- Ending character of the pratipadika (vowels/ consonants)

Vowel ending pratipadika

It follow the general pattern mostly. However, there is some Variation if we move across gender and categories of the pratipadikas as seen below .

Prathama Vibhakti[4]

There are three suffixes in prathamaVibhakti as su, au and jas. They areare used to express respectively ekavacana, dvivacanaandbahuvacana.

A' ending masculine.

For nominative singular (1-1), suffix is 'su', which is replaced by 'visarga' (:). There are five steps in replacing 'visarga' (su> s >ru> r >**:).** For example रामः,श्यामः,सर्वः,भारतः,एकः

In case of nominative and accusative dual (1-2/2-2), pratipadikaforms will be 'ौ'ending.[4] System will recognize 'ौ'. For example रामौ,शामौ,सर्वौ,एकौ. The system will isolate'ौ' and search for analysis by matching in the suffixdatabase.

In case of nominative plural (1-3), the pratipadikaforms will be 'ाः' ending. For Exampleरामाःश्यामः , भारताः,एकाः. The system will isolate 'ाः' and search for analysis by matching in the suffix database.

Vowel ending nominative singular/dual/plural patterns shows in table II.

V. DIFFERENT ALGORITHMS FOR SUBANTA ANALYSIS

There are various algorithms, developed for vibhakti identification by different sanskrit computation institute and experts.

A. Algorithm For Splitting The Word Basically Subanta Padas [8]

All the words within a given sentence is processed and splitsto find the root word and their corresponding suffix. If word isrAmbhyAm , it is mapped with all the suffix in noun

TABLE II.VOWEL ENDING NOMINATIVE SINGULAR/DUAL/PLURAL PATTERNS[4]

Vibhakti	Bases ending in	Subanta ending in	Change in Base (if any)	Examples
1-1	अ,इ,ई उ,ऊ,औ	ः	ः	रामः,हरिः भानुः,खलपूः
1-1	आ,ऋ	ा	ा	रमा,लता,पितृ
1-1	ऐ	ाः	ै	रामाः
1-2	अ	ौ		रामौ
1-2	आ	ो	ा	रमा,विश्वपा
1-2	इ	ी		हरी,मती
1-2	ई	यौ	ी	नद्यौ
1-2	उ	ू	ु	भानू
1-2	ऊ	वौ	ू	बाधू
1-2	ऋ	ारौ	ृ	पितरौ,धातारौ
1-2	ओ	ावौ	ो	गावौ
1-2	ऐ	ायौ	ै	रायौ
1-2	औ	ावौ	ौ	ग्लवौ
1-3	अ,आ	ाः	ा	रामाः,रमाः
1-3	इ	ायः		हरयः

database mapped suffixes are stored in list suffix database and their respective key values are stored in key list . If suffix database contains more than one element, then the suffix with maximum length is considered as final suffix and its key as final key value .From the key most significant digit gives the type of ending which is stored in k1.Next digit gives the gender, which is stored in k2 .If the value of k1 is 1 then after removing the mapped suffix, a is added to the word ,forming the root word . Similarly, depending upon the value of k1 either a,A,i,Ietc is added at the junction and root word is created .However if k1 is 1 but k2 is 2 , it means the word is in feminine gender , hence A is added to make root word. Table 2 summarizes the character to be added based on the values of k1 and k2. Again there may be more than one values in suffix database as in word rAmAbhyAm, AbhyAm is longest mapped suffix ,but there are multiple occurrence of AbhyAm in the database .Then the key values k1 and k2 are analyzed to find possible set of solutions.

rAmH= rama + h

रामः = राम: + प्रथमाएकवचन

Algorithm:

Step1: Input word w.

Step2: Match w in noun database to find suffix.

Step3: If match found

TABLE III: SUFFIX FOR A ENDING WORDS WITH

H (1111)	au (1112)	AH (1113)
Am (1121)	au (1122)	An (1123)
en (1131)	AbhyAm (1132)	ebhy H (1133)
At (1141)	AbhyAm (1142)	ebhy H (1143)
asya (1151)	yoH (1152)	ANAm (1153)
e (1161)	yoH (1162)	eShu (1163)

- Store the suffix to set suffix database
- Store their respective number in set key
- Filter suffix database and key so that it contains only longest suffix and its
- number respectively.
- Extract the first and second most significant digit from the number
- and store it in k1 and k2 respectively

Step4: Depending on the value of k1 and k2 split the word as root word and suffix. Add the character as per table (vowel/consonant table) at the end of the root word.

Step5: Display the possible results.

Step6: If more than one solution is obtain , map the root word in the dictionary.Word whose root word is found in the dictionary are the valid root words and final answer .

For example if the word is rAmHthen suffix database={'H'} and Key={1111} where k1=1 and k2 = 1 therefore it is identified as a ending word. If we spilt the word with respect to suffix H we get root word as rAm which becomes rAma after adding a and suffix obtained as H .This is simple example , let us examine some more cases When word= rAmAbhyAm then suffix database={ 'AbhyAm' 'AbhyAm' 'AbhyAm' 'yAm' 'yAm' 'AbhyAm' 'AbhyAm' 'AbhyAm' 'AbhyAm'}

Key={1232 1242 1252 2271 8271 1332 1342 1352 1132}

This is the case of multiple matched suffix where nine suffixes are mapped with the given word . Out of all these, suffixes with maximum length are selected .So now the set is reduced to suffix database=={ 'AbhyAm' 'AbhyAm' 'AbhyAm'

'AbhyAm' 'AbhyAm' 'AbhyAm' 'AbhyAm'}

Key={1232 1242 1252 1332 1342 1352 1132}

Logic selects the longest suffix and if k1 and k2 are analyzed ,it is observed that all the values are from a ending

word as k1=1 and gender can be masculine, feminine or neuter as k2 takes values 1,2 and 3 .Hence possible solution for the word

rAmAbhyAm is as follows:

rAmAbhyAm=rAma + AbhyAm
rAmAbhyAm=rAmA + AbhyAm

B. The Subanta Analysis System Works On A Subanta Rulebase [6]

This algorithm work on databases, Rulebase and example base. Databases of Puchuations, Avyayas and verbs are provided.

- Punctuation Database

This database contain all the punctuations ,so that they can be removed before processing.

- Verb Databases

This database is populated with all the verbs in the Sanskrit.

- Avyaya Database

All avyayas stored in this database which are used to compare with the word in input string.

- Example database

All complicated forms (which are not analyzed according to any rule) including those of some pronoun are stored the database.

- Rule database

The subanta patterns are stored in this database. This database analyzes those nouns which match a particular pattern from the rule base.

- Subanta eligibility
- check fixed databases (punctuations, avyayas, verbs)
  - If found tag
  - Else mark them SUBANTA
  - Check it in dictionary
    - If found store separately
    - Else start subanta processing
- check example database
  - If found tag and replace by correct matching
  - else continue
- Template search
  - Evaluate string as per set templates
  - If found obtain corresponding analysis
  - Display result as per analysis

Sample Illustration

Step 1:कृष्णचन्द्रःनामकश्चित्आसीत्।

Step 2:Recognition of verb-(आसीत्)

Step 3:Recognition of avyaya (कश्चित्)

Step 4: Recognition of subantas (कृष्णचन्द्रः)

Step 5: Analysis of Subantas

Step 6:कृष्णचन्द्रः] कृष्णचन्द्र) पुल्लिङ्ग + (सु, प्रथमा, एकवचन[[नाम_AV] [कश्चित्_AV] [आसीत्_VERB]

TABLE IV. COMPARISON OF APPROACHES FOR VIBHAKTI IDENTIFICATION

	Advantages	**Disadvantages**
Rule Based Approach	Based on linguistic theories. Adequate for languages with limited Resources. Does not require many computational Resources. Easy to perform error analysis. Easy to build an initial system.	Requires correct linguistic rules formulated by experts. Human Language Inconsistency (i.e. exceptions) Expensive to maintain and extend.
Statistic Based Approach	No linguistic knowledge required. Reduces the human resources cost. Easy to build. Easy to maintain . numerical knowledge required.	Requires high computational resources No linguistic background. Search cost is expensive
Knowledge Based Approach	Based on taxonomy of knowledge . Contains an inference engine.	They are quite expensive to produce due to the large amount of knowledge needed. Required expert in language.

VII. APPROACHES FOR VIBHAKTI IDENTIFICATION

In this section we discuss various vibhakti identification approaches proposed by various researchers.

A. Rule-Based approach

Rule-based approach is based on a rich repository of linguistic rules of language and dictionaries. In this, human experts specify a set of rules this is very expensive as linguistic experts need to work on it. The system consist of rules called grammar rules, lexicon and software programs to process the rules. These complex rule sets are used to perform Vibhakti identification. Subanta analyzer by sudhirmishara is example of Rule Based system [6].

B. Statistics Based Approach

The statistical approach is based on the statistical models whose parameters are derived from the analysis of language text corpora. It is a corpus based approach. Here corpora are maintained based on different probabilistic models. This approach is based on probability distribution. Building statistical translation models is a quick process, but the technology relies heavily on existing lingual corpora. Statistical approach is CPU-intensive and requires an extensive hardware configuration to run statisticmodels for average performance levels. statistical approaches can be used to remove the burden of rule induction from linguists to machines.[18] Google translate is an example of SMT[13].

C. *Knowledge-based Approach*

It follows the linguistic and computational instructions supplied to it by human researchers in linguistics and programming. It is the knowledge base that converts the source representation into an appropriate target representation before synthesizing into the target sentence.[18] This approach used to developed system on large scale of knowledge base. They are quite expensive to produce due to the largeamount of knowledge needed to accurately represent and compute sentences in different languages.

VIII. CONCLUSION

Vibhakti identification plays very important role because it is a pre-processing step which is very important step in any Natural Language Processing applications. One of the key task in identification of vibhakti is identifying the correct root word from its inflected form. Some Existing Subanta analyser systems were surveyed. Among those system Subantaanalyser system developed by sudhir k mishra and subhashchandra for subanta analyseris gives effective result with good accuracy. This system also has some limitations which can be overcome by addition sandhi splitting technique and improving database. The rule based approaches are based on hand written rules to analysis of Vibhakti. The comparison of various approaches for developing subanta analysis system shows rule based approach is most effective due to ease of building system. But it has some limitations like it requires linguistic rules formulated by experts. To enhance the performance as well as correctness of system hybrid approach can be used which combine two or more approaches.

REFERENCES

[1] Girish NathJha, Subash, Sudhir K. Mishra, Diwakar Mani, Diwakar Mishra, Manji Bhadra, Surjit K. Singh, " Inflectional Morphology Analyser for Sanskrit," L.S.I. at Hyderabad University, Hyderabad, pp-34 2008.

[2] PromilaBahadur, A.K Jain,D.S Chauhan, "EtranS-English to Sanskrit Machine Translation" ICWET 2012, Bombay, AC 2012.

[3] Huet, Gerard "Parsing Sanskrit by Computer" XIIth World Sanskrit Conference, Helsinki.

[4] Sudhir K. Mishra,"Sanskrit Karaka Analyzer for Machine Translation", a Ph. D. Thesis, SCSS JNU New Delhi, 2007.

[5] Ramanujan, P., 'Computer Processing Of Sanskrit', Computer Processing Of Asian Languages CALP-2, IIT Kanpur, India, 1992.

[6] Chandra, Subash, 'Machine Recognition and Morphological Analysis of Subanta padas ', M.Phil dissertation submitted to SCSS, JNU 2006.

[7] PawanGoyal, Vipul Arora, LaxmidharBehera, " Analysis Of Sanskrit Text: Parsing And Semantic Relations ", IIT Kanpur,, 2009.

[8] SmitaSelot, A.S. Zadgaonkar And Neeta Tripathi, " Subanta Pada Analyzer For Sanskrit ", , Oriental Journal Of Computer Science & Technology, May 15, 2010.

[9] AksharBharati, Amba Kulkarni, V Sheeba, "Building a Wide Coverage Sanskrit Morphological Analyzer: A PracticalmApproach", IIT Kanpur, 2009.

[10] Ramanujan, P., 'Computer Processing Of Sanskrit', Computer Processing Of Asian Languages CALP- 2, IIT Kanpur, India, 1992.

[11] *N.*Shailaja*"* Parser for Simple Sanskrit Sentences" M.Phil dissertation submitted to University of Hyderabad ,2009.

[12] Prof. Deepak Mane, AniketHirve," Study of Various Approaches in Machine Translation for Sanskrit Language", International Journal of Advancements in Research & Technology, Volume 2, Issue4, April-2013 383 ISSN 2278-7763.

[13] N. Murali, Dr. R.J. Ramasreee and Dr. K.V.R.K. Acharyulu." Kridanta Analysis For Sanskrit", International Journal on Natural Language Computing (IJNLC) Vol. 3, No.3, June 2014.

[14] N. Murali, Dr. R.J. Ramasreee and Dr. K.V.R.K. Acharyulu" Avyaya Analyzer: Analysis of Indeclinables using Finite State Transducers"International Journal of Computer Applications (0975 – 8887) Volume 38– No.6, January 2012.

[15] http://sanskrit.jnu.ac.in.

[16] http://www.baraha.com/BarahaIME.

[17] http://tdil.mit.gov.in/download/Desika.htm.

[18] http://www.translationdirectory.com