Expert Systems with Applications 38 (2011) 15084–15093

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

A TENGRAM method based part-of-speech tagging of multi-category words in Hindi language

J.P. Gupta [a], Devendra K. Tayal [b], Arti Gupta [c,*]

[a] *Jaypee Institute of Information Technology, Noida, Uttar Pradesh, India*
[b] *Department of Computer Engineering, GGSIP University, New Delhi, India*
[c] *Department of Computer Science & IT, Jaypee Institute of Information Technology, Noida, Uttar Pradesh, India*

ARTICLE INFO

Keywords:
Part-of-speech
Multi-category
Devanagari
TENGRAM
Discernibility matrix
Discernibility function
Reducts
Decision table
Rough sets
Decision rules
Condition attributes
Decision attribute

ABSTRACT

In this paper, we have dealt on the problem of part-of-speech tagging of multi-category words which appear within the sentences of Hindi language. Firstly, a Hindi tagger is proposed which provides part-of-speech tags developed using grammar of Hindi language. For this purpose, Hindi Devanagari alphabets are used and their Hindi transliteration is done within the proposed tagger. Thereafter, a Rules' based TENGRAM method is described with an illustrative example, which guides to disambiguate multi-category words within sentences of Hindi corpus. The rules generated in TENGRAM are the result of computation of discernibility matrices, discernibility functions and reducts. These computations have been generated from decision tables which are based on theory of Rough sets. Basically, a discernibility matrix helps in cutting down indiscernible condition attributes; a discernibility function has rows corresponding to each column in the discernibility matrix which develops reducts; and the reducts provide a minimal subset of attributes which preserve indiscernibility relation of decision tables and hence they generate the decision rules.

1. Introduction

Part-of-speech tagging (POS tagging or POST) (Araujo, 2002; Tapanainen & Voutilainen, 1994) is the process of marking up the words in a corpus corresponding to a particular part of speech, based on both its definition as well as its context (relationship with adjacent and related words in a sentence). POS provides significant information about a word and its adjacent members, clues on how a word is pronounced, word sense disambiguation, information extraction and question answering system. POS tagging is harder than just having a list of words and their parts-of-speech, because some words can represent more than one part-of-speech at different times. Such words are called multi-category words (MCW) and are ambiguous in nature. Our aim is to identify such ambiguous MCWs' occurring in Hindi corpus and to disambiguate them using Rules' based TENGRAM method which is proposed by us in the present paper.

In 2006, statistical algorithms such as Maximum Entropy Markov Model (MEMM) (Dalal, Nagraj, Sawant, & Shelke, 2006) was proposed which tagged previously unseen Hindi text using features based on the context, word, dictionary and corpus. These features captured the known information and then MEMM was applied to obtain the label sequence. Another, a comparative study of Unigram, Bigram, statistical Hidden Markov Model (HMM) and rule-based Brill's approaches for POS tagging was done in (Fahim, Naushad, & Mumit, 2007; Konchady, 2009). All these algorithms consider POS taggers for Hindi, but the tags used are confined to NN: noun, PRP: pronoun, VRB: verb, PREP: postposition etc., which are focused on English grammar. In this paper, we propose a Hindi tagger, having Hindi grammar which is based on Devanagari alphabets and their transliteration (elaborated in the next section). Devanagari is the script used for writing Hindi and the characters of Devanagari can be transcribed into English by replacing individual Devanagari letters with corresponding letters from the English language alphabets (Appendix A). Also, a Hindi professional can read text which is displayed with our fonts without much difficulty, which in turn helps in revitalization of Hindi, based on computational aspects of the language.

In the present paper, we use Rough sets (Pawlak, 1982, 1991; Polkowski, 2002; Polkowski & Skowron, 1998; Polkowski, Tsumoto, & Lin, 2000) to generate decision tables which are used to develop rules in the proposed TENGRAM method. The earlier approaches have mainly utilized the decision trees (Quinlan, 1987; Subramanian, Nosek, Raghunathan, & Kanitkar, 1992) for the same purpose. Decision tree solves simpler problems better but when number of actions is large, many combinations of conditions exist, and there is a risk of ambiguities and omissions. For such complex cases decision tables are preferred over decision trees. Rough sets

* Corresponding author. Mobile: +91 9313519476.
E-mail addresses: arti.gupta@jiit.ac.in, frds_4_arti@yahoo.com (A. Gupta).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.05.036

(Demri & Orlowska, 2002; Lin & Cercone, 1997; Orlowska, 1997; Sivanandam & Deepa, 2008) used by us in the proposed model, resolve imprecision approximately by expressing quantitative concepts and does feature selection and reduction through computation of discernibility matrix and discernibility function (Skowron & Rauszer, 1992) which result in the formulation of reducts of attributes that are efficient for rules formulation and correction. Further enhancement in the rule-base is done by eliminating inconsistent rules as discussed in detail in the later sections of this paper.

2. Background and related work

This section is organized as follows: Section 2.1 discuuses about literature survey in Hindi. Section 2.1.1.describes existing intermediate tools and models for initial processing of POST in Hindi, Section 2.1.2 describes existing jargon for POST in Hindi. The terminology for Hindi that is used by us in the present paper is elaborated in Section 2.1.3. Literature survey of Rough sets (Section 2.2) describes with the steps to be followed for rules generation using TENGRAM method (Section 3.2, Steps I–IV), along with inconsistent rules elimination (Section 3.2, Step V).

2.1. Literature survey of Hindi

2.1.1. Existing tools and models for POST in Hindi

Hindi is a morphologically rich Indian language and hence needs analyzing tools for harnessing its structural information. Some intermediate tools developed for initial processing of structural information in Hindi are Stemmer and Morphological-Analyser (Shrivastava, Agrawal, Mohapatra, Singh, & Bhattacharya, 2005). Stemmer provides the root, suffix and grammatical category of an input word. Morphological-Analyser gives detailed analysis of the word based on its inflection and context of its usage. Both, Stemmer and Morphological-Analyzer tools are unable to produce efficient results for large number of words, either because words are not present in the word list or due to some spelling errors. Models such as SUFFIX (Rao & Yarowsky, 2007) uses Naïve-Bayes assumption, but for Out-of-Vocabulary (OOV) words. SUFFIX assigns a tag that maximizes the likelihood of that tag, given a fixed length suffix of the OOV word.[1]

2.1.2. Existing jargon for POST in Hindi

In their paper (Ray, Harish, Sarkar, & Basu, 2003) proposed an algorithm for local word grouping to untangle fixed word order (FWO) dependencies in Hindi sentences. Hindi being a free order language, FWO group extraction decreases load on the free order parser and the POST is an essential requirement for local word grouping. The authors denote the set of all relevant and existing POS in Hindi as C, where $C = \{n, v, a, j, c, pp, q, y\}$; n: noun or pronoun, v: verb, a: adverb, j: adjective, c: conjunction, pp: postposition, q: qualifier, y: other parts-of-speech. FOLLOW (Ray et al., 2003), a binary relationship and a subset of $C * C$, puts restriction on whether a POS tag can follow another POS tag in a Hindi sentence. FOLLOW is constructed by empirical observation of Hindi and has been hand-tested and found to be correct on sentences of varying complexity.

Table 1 depicts FOLLOW (x) = {set of all POS tags that can follow x in a Hindi sentence} or FOLLOW $(x) = \{y$: (Row x, Column y) is 1}. The rows of FOLLOW can be illustrated as- (i) In row1, current word has noun/pronoun (n) as its POS, therefore the words following to (n) can have any of the {n, j, v, a, c, pp, q, y}as POS tags within the context. (ii) In row2, current POS is adjective (j), therefore words following to (j) within a given context have possibly {n, j, c, q} as POS tag; no other tag can be assigned. The other rows of the Table 1 can be illustrated similarly.

Table 1
FOLLOW relation in Hindi.

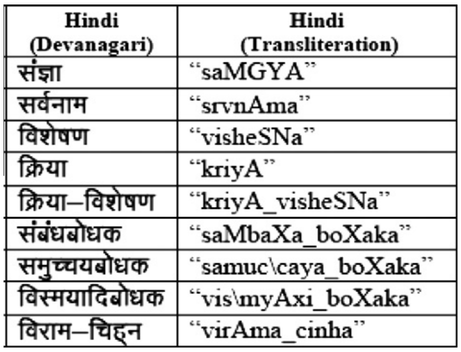
	n	j	v	a	c	pp	q	y
n	1	1	1	1	1	1	1	1
j	1	1	0	0	1	0	1	0
v	1	1	1	1	1	1	1	1
a	1	1	1	1	1	1	1	1
c	1	1	1	1	1	0	1	1
pp	1	1	1	1	1	1	1	1
q	0	1	0	1	0	0	1	0
y	1	1	1	1	1	0	1	1

2.1.3. Some more terminology

Some taggers already exist for Hindi, but while deciding their POS tags, the English grammar has always been considered as the base. Some of these defined tags for Hindi are- NN: Noun (e.g. boy, river, thought, hardness), PRP: Pronoun (e.g. who, that, he), QW: Question Words (e.g. what, who), RB: Adverb (e.g. slowly slowly, fast), INIF: Intensifier (e.g. too much, much more), NEG: Negative (e.g. no, not), CC: Conjucts (e.g. and, or), QF: Quantifier (e.g. more, little, all, much), etc. Our focus in this paper is to use Hindi grammar for Hindi words, since Hindi language uses a different word-order (subject-object-verb) over the English language word-order (subject-verb-object). In other words, in Hindi, verbs are placed at the end of a sentence. Also, Hindi uses postpositions instead of prepositions, which are similar to prepositions except that they are written after the noun. In addition, Hindi transliteration done in this paper helps non-Hindi readers to pronounce Hindi alphabets into the known (English language) character format.

In this paper we define a Hindi tagger with POS tags, available within the Hindi grammar described in (Visvendru, 1993). Hindi Devanagari alphabets are used to define POS and their Hindi transliteration is done using English alphabets (both lower and upper cases) as shown in Figs. 1 and 2. List of Hindi Devnagri alphabets with Hindi transliteration is available in Appendix A.

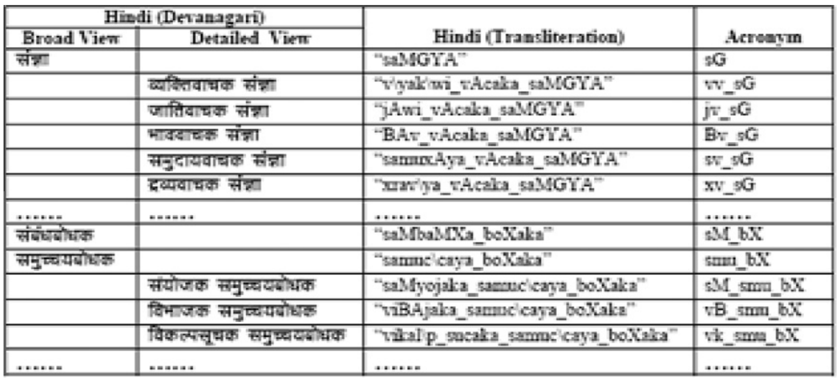
So, C: set of POS tags in Hindi, as stated in the jargon for POST in Hindi (Section 2.1.2) is redefined as C' in Fig. 1. C' is defined as, C' = {"saMGYA"/noun, "srvnAma"/pronoun, "visheSNa"/adjective, "kriyA"/verb, "kriyA_visheSNa"/adverb, "saMbaXa_boXaka"/postposition, "samuc\caya_boXaka"/conjunction, "vis\myAXi_boXaka"/interjection, "virAma_cinha"/punctuation}. It is important to note that the words enclosed within " " are especially Hindi



Hindi (Devanagari)	Hindi (Transliteration)
संज्ञा	"saMGYA"
सर्वनाम	"srvnAma"
विशेषण	"visheSNa"
क्रिया	"kriyA"
क्रिया–विशेषण	"kriyA_visheSNa"
संबंधबोधक	"saMbaXa_boXaka"
समुच्चयबोधक	"samuc\caya_boXaka"
विस्मयादिबोधक	"vis\myAxi_boXaka"
विराम–चिह्न	"virAma_cinha"

Fig. 1. Broad level POS tags in Devanagari and their Hindi transliteration.

[1] A Part of Speech Tagger for Indian Languages http://shiva.iiit.ac.in/SPSAL2007/iiit_tagset_guidelines.pdf. Hindi Pad http://www.softpedia.com/progDownload/HindiPad-Download-16310.html. Part-of-Speech Tutorial http://alias-i.com/lingpipe/demos/tutorial/posTags/read-me.html. Rough Sets: A Tutorial http://folli.loria.fr/cds/1999/library/pdf/skowron.pdf.



Hindi (Devanagari)		Hindi (Transliteration)	Acronym
Broad View	Detailed View		
संज्ञा		"saMGYA"	sG
	व्यक्तिवाचक संज्ञा	"v\yak\wi_vAcaka_saMGYA"	vv_sG
	जातिवाचक संज्ञा	"jAwi_vAcaka_saMGYA"	jv_sG
	भाववाचक संज्ञा	"BAv_vAcaka_saMGYA"	Bv_sG
	समुदायवाचक संज्ञा	"samuxAya_vAcaka_saMGYA"	sv_sG
	द्रव्यवाचक संज्ञा	"xrav\ya_vAcaka_saMGYA"	xv_sG
......	......	......	......
संबंधबोधक		"saMbaMXa_boXaka"	sM_bX
समुच्चयबोधक		"samuc\caya_boXaka"	smu_bX
	संयोजक समुच्चयबोधक	"saMyojaka_samuc\caya_boXaka"	sM_smu_bX
	विभाजक समुच्चयबोधक	"viBAjaka_samuc\caya_boXaka"	vB_smu_bX
	विकल्पसूचक समुच्चयबोधक	"vikal\p_sucaka_samuc\caya_boXaka"	vk_smu_bX
......	......	......	......

Fig. 2. Sample of detailed Hindi POS tags – Devanagari, Hindi transliteration and acronyms.

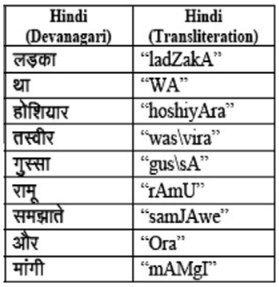
transliterated words and are case sensitive (upper and lower case letters of English).

Fig. 2 depicts a sample of detailed POS tags defined in Hindi grammar using Devanagari, along with their Hindi transliteration and acronyms. For an instance, "saMGYA"/"sG" is a broad POS tag (Fig. 1) and its detailed tags are "v\yak\wi_vAcaka_saMGYA"/ "vv_sG", "jAwi_vAcaka_saMGYA"/"jv_sG", "BAv_vAcaka_saMGYA"/ "Bv_sG", "samuxAya_vAcaka_saMGYA"/"sv_sG" and "xrav\ya_ vAcaka_saMGYA"/"xv_sG". On the other hand, some tags like "saMbaMXa_boXaka"/"sM_bX" have no further detailing and hence have no detailed view. Similarly, other tags in Fig. 2 can be illustrated.

Once, POS tags have been defined in Hindi syntax along with their acronyms, we propose a modified FOLLOW relation as shown in Table 2. Table 2 is generated by modifying Table 1 of (Ray et al., 2003), using language based heuristic rules of (Tapanainen & Voutilainen, 1994) and Hindi grammar from (Visvendru, 1993). In this paper we describe "saMGYA"/"sG", "srvnAma"/"sr", "visheSNa"/"vN", "kriyA"/"ky", "kriyA_visheSNa"/"kvN", "saMbaMxa_boXaka"/"sM_ bX", "samuc\caya_boXaka"/"smu_bX", "vis\myAXi_boXaka"/ "vmy_bX", "virAma_cinha"/"vr_cn", as two-dimensional attributes having values as 1 or 0, depending on whether FOLLOW relation is observed or not.

Once POS tags in the Hindi grammar and the modified FOLLOW relation are defined in the proposed Hindi tagger, then the Devanagari alphabets are used to construct Hindi corpus.

Sample words from Hindi corpus are shown in Fig. 3 which includes some of the मात्राएँ "mAtrAeM" supported in Hindi like आ "A",ई"E",उ"u" etc.



Hindi (Devanagari)	Hindi (Transliteration)
लड़का	"ladZakA"
था	"WA"
होशियार	"hoshiyAra"
तस्वीर	"was\vira"
गुस्सा	"gus\sA"
रामू	"rAmU"
समझाते	"samJAwe"
और	"Ora"
मांगी	"mAMgI"

Fig. 3. Sample words of Hindi transliteration to Devanagari from Hindi training corpus.

2.2. Rough sets- towards POS tagging

Rough sets (Pawlak, 1982, 1991) is a mathematical tool introduced by Pawlak in 1980's. It concerns with the analysis and modeling of classification and decision problems involving vagueness, impreciseness and uncertainity or partial information. Rough sets offers two kinds of knowledge representations called information system and decision system.

An information system (IS) is the most basic kind of knowledge that consists of a set of objects where each object is a collection of attributes values. An IS is an ordered pair, $\mathcal{A} = (U, A)$, where U is a nonempty finite set called universe and A is a nonempty finite set of attributes. Each attribute $a \in A$ is a total function a: $U \rightarrow V_a$, where V_a is the set of values of a, called range of a. The element in universe is referred as objects.

A decision system (DS), $\mathcal{A} = (U, A, \{d\})$ is an IS, for which the attributes are separated into disjoint sets of condition attributes A and decision attributes d, $(A \cap \{d\} = \phi)$.

In this paper, our focus is on the decision system which enables rules' generation to determine POS tags of MCWs' within Hindi corpus.

3. Methodology

3.1. Framework

In this section we provide an overall flow of our research work. We use the Tool 'KrutiPAD 010' http://www.softpedia.com/

Table 2
Modified FOLLOW relation for broad POS tags in Hindi.

	"sG"	"sr"	"vN"	"ky"	"kvN"	"sM_bX"	"smu_bX"	"vmy_bX"	"vr_cn"
"sG"	1	1	1	1	1	1	1	0	1
"sr"	1	1	1	1	1	1	1	0	0
"vN"	1	1	1	0	0	1	0	0	1
"ky"	1	1	1	1	1	1	1	0	1
"kvN"	1	1	1	1	1	1	1	0	1
"sM_bX"	1	1	1	1	1	1	0	0	0
"smu_bX"	1	1	1	1	1	1	1	0	0
"vmy_bX"	1	1	1	1	1	1	0	0	0
"vr_cn"	1	1	1	1	1	1	0	1	1

progDownload/HindiPad-Download-16310.html, a Hindi editor to enable us to write Hindi alphabets, once Hindi fonts are installed on the computer system.

The work-flow starts from preparing a list of Hindi alphabets in Devanagari which once transliterated, enables construction of: (i) POS tagger in Hindi grammar, (ii) Hindi training corpus. The POS tagger proposed here is then applied over the built Hindi corpus, converting that corpus into manual-tagged corpus. From the tagged corpus, we have identified multi-category words through implementation in '*C*'- language. Further, to disambiguate MCWs', we have applied TENGRAM method (Section 3.2). Firstly, TENGRAM prepares decision table for each MCW having condition and decision attributes. After that, TENGRAM reduces the decision table to discernibility matrix (to avoid redundancy of indiscernible objects), discernibility function is then extracted from discernibility matrix (rows of discernibility function corresponds to columns of discernibility matrix). Later on, TENGRAM computes reducts from discernibility function which then generates decision or dependency rules. If decision rules generated above are found to be inconsistent then they must be eliminated from TENGRAM, otherwise the rules are left unchanged.

Hence, we obtain processed Hindi corpus with the rules identified for MCWs' in the TENGRAM. Thus generation of consistent decision rules can be used as-it-is in any other tested Hindi corpus.

Fig. 4 describes the framework of our proposed work applied to Hindi language.

3.2. TENGRAM method

TENGRAM method uses rough set features for rules generation (Qian & Zheng, 2004) of POST in Hindi language. TENGRAM consists of the following four important steps:

(i) Formulation of TENGRAM decision table.
(ii) Computation of TENGRAM discernibility matrix, for each decision class.
(iii) Computation of TENGRAM discernibility function, for each object of the decision table belonging to a certain decision class.
(iv) Generation of TENGRAM dependency rul*e*s from the reducts generated.

At the end, we have added one more step, i.e. elimination of TENGRAM inconsistent rules for enhancing accuracy of decision rules generated.

The above mentioned steps are detailed as follows:

Step I. Formulation of TENGRAM decision table

First step of TENGRAM method consists of formulation of decision table having POS tags of five consecutive words (before a particular MCW) and POS tags of five other consecutive words (after the same MCW) within Hindi sentences. They are called as the condition attributes. Also, POS tag of the MCW under consideration is called as a decision attribute. The TENGRAM decision table is as shown in Fig. 5.

In other words, TENGRAM decision table is a decision system (D_T), represented as $D_T = (U, C_A \cup D_A)$, where $U = \{s_1, s_2, \ldots, s_n\}$ is a collection of sentences s_i,with MCW in Hindi corpus.

C_A, $D_A \subset A$, where $A = \{c_{-5}, c_{-4}, \ldots, c_{-1}, c_0, c_1, c_2, \ldots, c_5\}$ is an attribute (column) set, and $c_j (j \in [-5, 5])$ is an attribute.

$C_A = \{c_{-5}, c_{-4}, \ldots, c_{-1}, c_1, c_2, \ldots, c_5\}$ is a subset of condition attributes, made up of five POS tags before the considered MCW and five POS tags after that MCW.

$D_A = \{c_0\}$ is the subset of decision attributes; c_0 is the current POS tag of the MCW.

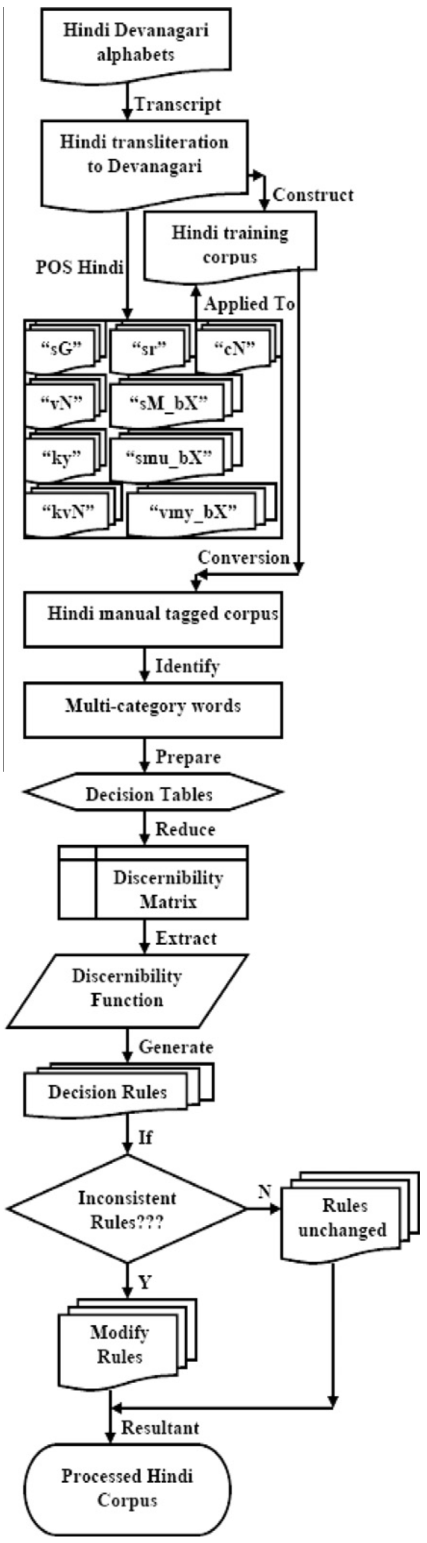
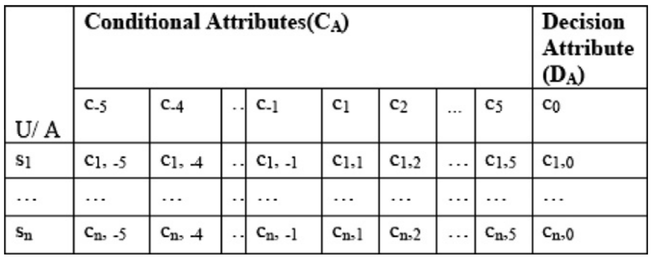


Fig. 4. Research framework of POST and TENGRAM method for Hindi language.

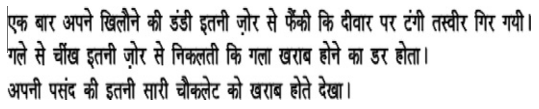
Similarly, the rest of the entries in the Fig. 5 can be explained.

It may be possible that some condition attributes (farthest from decision attribute on either side) may not be assigned any of the POS tags since the words corresponding to these attributes donot occur (i.e. null) within Hindi sentences. Such condition attributes are assigned null value (ϕ) and is explained with an example below.



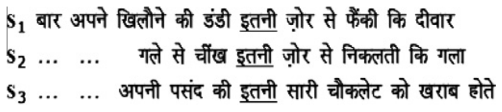
U/ A	Conditional Attributes(C_A)								Decision Attribute (D_A)
	c_{-5}	c_{-4}	..	c_{-1}	c_1	c_2	…	c_5	c_0
s_1	$c_{1,-5}$	$c_{1,-4}$	..	$c_{1,-1}$	$c_{1,1}$	$c_{1,2}$	…	$c_{1,5}$	$c_{1,0}$
…	…	…	..	…	…	…	…	…	…
s_n	$c_{n,-5}$	$c_{n,-4}$	..	$c_{n,-1}$	$c_{n,1}$	$c_{n,2}$	…	$c_{n,5}$	$c_{n,0}$

Fig. 5. TENGRAM decision table for POST of MCW.



एक बार अपने खिलौने की डंडी इतनी ज़ोर से फैंकी कि दीवार पर टंगी तस्वीर गिर गयी।
गले से चींख इतनी ज़ोर से निकलती कि गला खराब होने का डर होता।
अपनी पसंद की इतनी सारी चौकलेट को खराब होते देखा।

Fig. 6. Sentences with Hindi word "iwanI" इतनी from Hindi corpus.



s_1 बार अपने खिलौने की डंडी इतनी ज़ोर से फैंकी कि दीवार
s_2 … … गले से चींख इतनी ज़ोर से निकलती कि गला
s_3 … … अपनी पसंद की इतनी सारी चौकलेट को खराब होते

Fig. 7. Sentences (s_1, s_2, s_3) with Hindi word "iwanI"इतनी.

Example 1. Consider sentences from Hindi corpus having Hindi word "iwanI"इतनी as shown in Fig. 6.

Based on TENGRAM, above sentences of Fig. 6 are treated as s_1, s_2, s_3 as given in Fig. 7.

Here, we consider only five consecutive words on either side of the word इतनी. If some boundary words donot exist they are left blank (...), can be seen in the above sentences s_2 and s_3 both.

Hindi transliteration of s_1,s_2 and s_3 along with their POS tags (based on context information) and acronyms are shown in Table 3.

In Table 3, POS tags are computed, based on the context information. The POS tags which are used here are: "Bv_sG", "jv_SG", "xv_sG", "gN_sG" as types of "saMGYA"/noun; "njv_sr" as type of "srvnAma"/pronoun; "prv_vN", "gN_vN", "sKv_vN" as types of "visheSNa"/adjective; "mu_ky", "vN_ky" as types of "kriyA"/verb where "mu_ky" is the main verb of a sentence and "vN_ky" is a verb used as an adjective; "sM_bX" is a "saMbaMXa_boXaka"/postposition; "vB_smu_bX" is a type of "samuc\cya_boXaka"/conjunction. It can be seen that "iwanI" is a multi-category word, based on the usage of its context. So, firstly, we construct a decision table for MCW "iwanI" as shown in Table 4.

Table 4 has condition attributes as POS tags (acronyms) of adjacent words (adjacent to word "iwanI") on both sides of "iwanI", within sentences of Hindi corpus (refer Table 3). Also, decision attribute is defined as POS tags (acronyms) of the MCW "iwanI". Sentence s_1 has five previous POS tags as c_{-5} = "Bv_sG", c_{-4} = "njv_sr", c_{-3} = "jv_sG", c_{-2} = "sM_bX", c_{-1} = "jv_sG"; five next tags as c_1 = "Bv_sG", c_2 = "sM_bX", c_3 = "mu_ky", c_4 = "vB_smu_bX", c_5 = "jv_sG"; and POS tag of "iwanI" (within s_1) as c_0 = "prv_vN". Similarly, POST for s_2 and s_3 are defined in the Table 4. It is also seen that c_{-5} and c_{-4} in s_2 and s_3 are ϕ, since words corresponding to these tags donot exist within s_2 and s_3 respectively.

Step II. Computation of TENGRAM discernibility matrix

TENGRAM decision table constructed above has so many indiscernible objects which must be reduced by cutting down its condition attributes. So, we compute TENGRAM discernibility matrix (TDM) which is denoted as:

$M(D_T) = [m_{ij}]_{n*n}$, where n: number of sentences with MCW, m_{ij}, the set of attributes which discerns between objects x_i and x_j and also discerns the decision attributes c_0, defined as: $m_{ij} = \{c_k \in A: c_k(x_i) \neq c_k(x_j)$, for $k = -5, -4, \ldots, -1, 1, \ldots, 5$ and $1 \leqslant i,j \leqslant n\}$, where $c_0(x_i) \neq c_0(x_j)$.

TDM is symmetric and $c_{ii} = \phi$, for each $i = 1,2,\ldots,n$,. Hence, there exists only half of elements in TDM in comparison to D_T.

We now compute TDM for word "iwanI" (discussed in Example 1) in Table 5.

Values in Table 5 are computed from decision table constructed in Table 4. Here we have, discernibility matrix of order 3*3 with $c_{ii} = \phi$, i = 1, 2, 3 and indiscernible decision attributes for s_1 and s_2 (since $c_0(x_1) = c_0(x_2)$ = "prv_vN" (Table 4)). So, only discernible objects left are for two cells (column s_1 and row s_3; column s_2 and row s_3 respectively). For obtaining values in these cells, we compare c_k for $k = -3, -2, -1, 1, 2, \ldots, 5$ (since c_{-5} and c_{-4} are both ϕ for s_2 and s_3 respectively (Table 4)). It is observed that when column s_1 and row s_3 are compared, then discernible objects are c_{-3}, c_{-2}, c_{-1}, c_1, c_2, c_3, c_4, c_5. Similarly, in the resultant discernibility matrix values for column s_2 and row s_3 are obtained, as shown in Table 5.

Step III. Computation of TENGRAM discernibility function

We compute TENGRAM discernibility function (TDF) as $f(M(D_T))$ based on the TDM as a Boolean function of m variables $c_1*, \ldots, c_m*$(corresponding to $c_1, \ldots, c_m$) defined as:

$f(M(D_T))(c_1*, \ldots, c_m*) = \wedge\ \{\vee\, m_{ij}*\ :\ 1 \leqslant i,j \leqslant n, m_{ij} \neq \phi\}$, where $m_{ij}* = \{c* : c \in m_{ij}\}$.

Each row in the TDF corresponds to one column in the TDM, for each object x_i of the decision table belonging to a certain decision class.

We now compute TDF for word "iwanI" using Table 5. In this table, column 1 has only one value (corresponding to row s_3), so the computation of $\wedge$ operator becomes trivial, and only the value

Table 3
Hindi transliteration (H_Trans), POS tagging (P_Tag) and acronymn (Acr) of words in sentences (S#)- s_1, s_2 and s_3 respectively.

S#	H_Trans	P_ Tag	Acr
s_1	"bAra"	"BAv_vAcaka_saMGYA"	"Bv_sG"
	"apane"	"nija_vAcaka_srvnAma"	"njv_sr"
	"KilOne"	"jAwi_vAcaka_saMGYA"	"jv_sG"
	"kI"	"saMbaMXa_boXaka"	"sM_bX"
	"daMdI"	"jAwi_vAcaka_saMGYA"	
	"iwanI"	"parimANa_vAcaka_visheSNa"	"prv_vN"
	"jZora"	"BAv_vAcaka_saMGYA"	
	"se"	"saMbaMXa_boXaka"	
	"PeMkI"	"muK\ya_kriyA"	"mu_ky"
	"ki"	"viBAjaka_samuc\cya_boXaka"	"vB_smu_bX"
	"xIvAra"	"jAwi_vAcaka_saMGYA"	
s_2	"gale"	"jAwi_vAcaka_saMGYA"	
	"se"	"saMbaMXa_boXaka"	
	"cIMKa"	"BAv_vAcaka_saMGYA"	
	"iwanI"	"parimANa_vAcaka_visheSNa"	
	"jZora"	"BAv_vAcaka_saMGYA"	
	"se"	"saMbaMXa_boXaka"	
	"nikalawI"	"muK\ya_kriyA"	
	"ki"	"viBAjaka_samuc\cya_boXaka"	
	"gala"	"jAwi_vAcaka_saMGYA"	
s_3	"apanI"	"nija_vAcaka_srvnAma"	
	"pasaMxa"	"guNa_vAcaka_visheSNa"	"gN_vN"
	"kI"	"saMbaMXa_boXaka"	
	"iwanI"	"saMbaMXa_boXaka"	
	"sArI"	"saMK\yA_vAcaka_visheSNa"	"sKv_vN"
	"cOkaleta"	"xrav\ya_vAcaka_saMGYA"	"xv_sG"
	"ko"	"saMbaMXa_boXaka"	
	"KarAba"	"guNa_vAcaka_saMGYA"	"gN_sG"
	"howe"	"visheSNa_kriyA"	"vN_ky"

Table 4
TENGRAM decision table for MCW “iwanI”.

U/A	Conditional Attributes (C_A)										Decision Attribute (D_A)
	c_{-5}	c_{-4}	c_{-3}	c_{-2}	c_{-1}	c_1	c_2	c_3	c_4	c_5	c_0
s_1	“Bv_sG”	“njv_sr”	“jv_sG”	“sM_bX”	“jv_sG”	“Bv_sG”	“sM_bX”	“mu_ky”	“vB_smu_bX”	“jv_sG”	“prv_vN”
s_2	ϕ	ϕ	“jv_sG”	“sM_bX”	“Bv_sG”	“Bv_sG”	“sM_bX”	“mu_ky”	“vB_smu_bX”	“jv_sG”	“prv_vN”
s_3	ϕ	ϕ	“njv_sr”	“gN_vN”	“sM_bX”	“sKv_vN”	“Xv_sG”	“sM_bX”	“gN_sG”	“vN_ky”	“sM_bX”

Table 5
TENGRAM discernibility matrix for MCW “iwanI”.

	s_1	s_2	s_3
s_1	ϕ		
s_2	ϕ	ϕ	
s_3	$c_{-3}, c_{-2}, c_{-1}, c_1, c_2, c_3, c_4, c_5$	$c_{-3}, c_{-2}, c_{-1}, c_1, c_2, c_3, c_4, c_5$	ϕ

corresponding to $(c_{-3} \vee c_{-2} \vee c_{-1} \vee c_1 \vee c_2 \vee c_3 \vee c_4 \vee c_5)$ will occur. Similarly, column 2 and column 3 are computed.

Step IV. Generation of TENGRAM dependency rules

TENGRAM dependency rules are also known as correction or decision rules. These rules are generated directly for each generated reduct, where reducts stand for minimal subsets of condition attributes which preserve indiscernibility relation. A reduct is represented as RED(A), and is obtained from TENGRAM discernibility function $f(M(D_T))$.

It can be noted that minimum subset of attributes (i.e. reducts) for word “iwanI” is $(c_{-3} \vee c_{-2} \vee c_{-1} \vee c_1 \vee c_2 \vee c_3 \vee c_4 \vee c_5)$. Hence, based on this reduct, we can generate rules for word “iwanI”. The pseudo code for generation of dependency rules for word “iwanI” is given in Fig. 8.

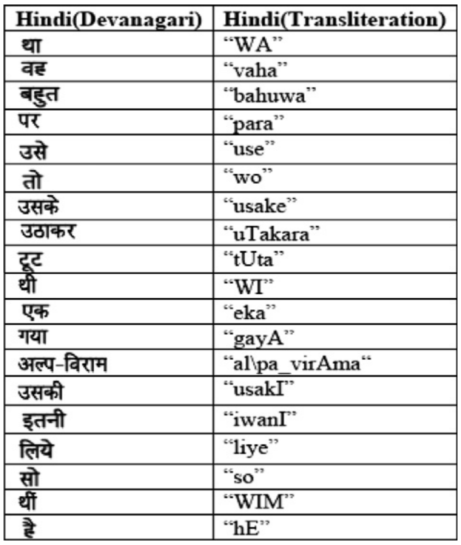
Step V. Elimination of TENGRAM inconsistent rules

Once rules are generated through TENGRAM method, then it is essential to check the consistency of these rules.

TENGRAM rules are formulated as: if (condition is x) then (result is y).

Suppose we have a rule, if (condition is x') then (result is y'), where $x = x'$.

In this case, it is essential to have $y = y'$ for rule to be consistent, otherwise rule is inconsistent. Examples of elimination of inconsistent rules are stated in the next section.



Hindi(Devanagari)	Hindi(Transliteration)
था	“WA”
वह	“vaha”
बहुत	“bahuwa”
पर	“para”
उसे	“use”
तो	“wo”
उसके	“usake”
उठाकर	“uTakara”
टूट	“tUta”
थी	“WI”
एक	“eka”
गया	“gayA”
अल्प-विराम	“al\pa_virAma“
उसकी	“usakI”
इतनी	“iwanI”
लिये	“liye”
सो	“so”
थीं	“WIM”
है	“hE”

Fig. 9. List of the identified MCWs’ within Hindi corpus.

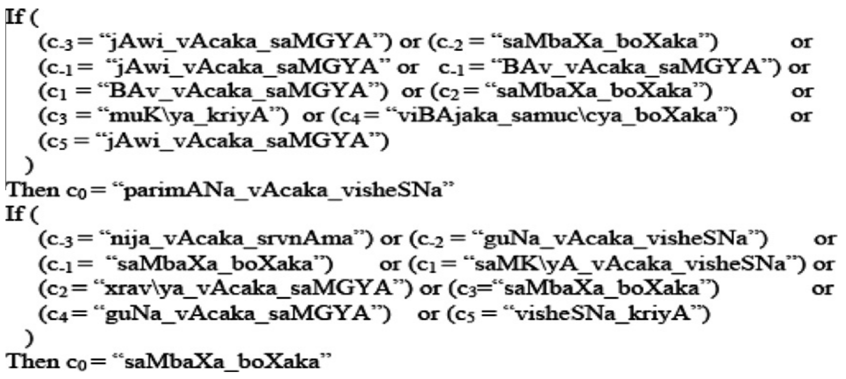
4. Results

4.1. Some identified multi-category words in Hindi

In our present work, we have identified 19 multi-category words and they appear 116 times in our developed Hindi corpus. List of identified MCWs’ are shown in the Fig. 9.

4.2. Rules generation for each MCW in Hindi

For implementation purpose in ‘C’-language, we have used a subset of TENGRAM method, which considers three words viz. the MCW and two other words (immediate adjacent neighbours



```
If (
    ($c_{-3}$ = “jAwi_vAcaka_saMGYA”) or ($c_{-2}$ = “saMbaXa_boXaka”)          or
    ($c_{-1}$ = “jAwi_vAcaka_saMGYA” or  $c_{-1}$ = “BAv_vAcaka_saMGYA”) or
    ($c_1$ = “BAv_vAcaka_saMGYA”) or ($c_2$ = “saMbaXa_boXaka”)            or
    ($c_3$ = “muK\ya_kriyA”) or ($c_4$ = “viBAjaka_samuc\cya_boXaka”)      or
    ($c_5$ = “jAwi_vAcaka_saMGYA”)
  )
Then $c_0$ = “parimANa_vAcaka_visheSNa”
If (
    ($c_{-3}$ = “nija_vAcaka_srvnAma”) or ($c_{-2}$ = “guNa_vAcaka_visheSNa”)      or
    ($c_{-1}$ = “saMbaXa_boXaka”)      or ($c_1$ = “saMK\yA_vAcaka_visheSNa”) or
    ($c_2$ = “xrav\ya_vAcaka_saMGYA”) or ($c_3$=“saMbaXa_boXaka”)            or
    ($c_4$ = “guNa_vAcaka_saMGYA”)    or ($c_5$ = “visheSNa_kriyA”)
  )
Then $c_0$ = “saMbaXa_boXaka”
```

Fig. 8. Generation of rules for MCW “iwanI”.

Table 6
Decision rules for identified MCWs' using immediate adjacent neighbours.

MCW#	Rule#	MCW	Prev_tag	Next_tag	Curr_tag
1		"WA"			
	R1.1		"jv_sG"		"mu_ky"
	R1.2		"prv_kvN"		"mu_ky"
	R1.3		"sM_bX"		"mu_ky"
	R1.4		"gN_vN"		"mu_ky"
	R1.5		"mu_ky"		"amu_ky"
	R1.6		"amu_ky"		"amu_ky"
2		"vaha"			
	R2.1			"Bv_sG"	"apv_sr"
	R2.2			"nSv_sr"	"apv_sr"
	R2.3			"gN_vN"	"apv_sr"
	R2.4			"sMv_vN"	"apv_sr"
	R2.5			"vN_ky"	"apv_sr"
	R2.6			"kvN_ky"	"nSv_sr"
	R2.7			"jv_sG"	"sMv_vN"
3		"bahuwa"			
	R3.1		"sM_bX"		"sM_bX"
	R3.2			"mu_ky"	"prv_kvN"
	R3.3		"apv_sr"		"prv_vN"
	R3.4		"vv_sG"		"prv_vN"
	R3.5		"sy_smu_bX"		"prv_vN"
	R3.6		"vB_smu_bX"	"gN_vN"	"prv_vN"
4		"para"			
	R4.1		"jv_sG"		"sM_bX"
	R4.2		"sG_ky"		"sM_bX"
	R4.3		"ap_cn"		"vB_smu_bX"
	R4.4		"amu_ky"		"vB_smu_bX"
5		"use"			
	R5.1			"Bv_sG"	"sMv_vN"
	R5.2			"mu_ky"	"apv_sr"
6		"wo"			
	R6.1		"ap_cn"		"vB_smu_bX"
	R6.2		"amu_ky"		"vB_smu_bX"
	R6.3		"sWv_kvN"		"vB_smu_bX"
	R6.4		"jv_sG"		"vB_smu_bX"
	R6.5		"vv_sG"		"vB_smu_bX"
	R6.6		"mu_ky"	"sWv_kvN"	"Bv_sG"
	R6.7		"mu_ky"	"prv_kvN"	"Bv_sG"
7		"usake"			
	R7.1			"jv_sG"	"sMv_vN"
	R7.2			"mu_ky"	"apv_sr"
8		"uTAkara"			
	R8.1		"ap_cn"		"kvN_ky"
	R8.2		"jv_sG"		"kvN_ky"
	R8.3		"sM_bX"		"sM_bX"
9		"tUta"			
	R9.1		"jv_sG"		"gN_vN"
	R9.2		"kv_kvN"		"mu_ky"
10		"WI"			
	R10.1		"amu_ky"		"amu_ky"
	R10.2		"nSv_sr"	"vB_smu_bX"	"mu_ky"
11		"eka"			
	R11.1			"jv_sG"	"sKv_vN"
	R11.2			"Bv_sG"	"prv_vN"
12		"gayA"			
	R12.1		"mu_ky"		"amu_ky"
	R12.2		"gN_vN"		"mu_ky"
	R12.3		"vB_smu_bX"		"mu_ky"
13		"al\pa_virAma" (-)			
	R13.1		"amu_ky"		"nX_cn"
	R13.2		"vN_ky"		"nX_cn"
	R13.3		"gN_vN"	"gN_vN"	"yk_cn"

Table 6 (*continued*)

MCW#	Rule#	MCW	Prev_tag	Next_tag	Curr_tag
14		“usakI”			
	R14.1			“jv_sG”	“sKv_vN”
	R14.2			“sWv_kvN”	“apv_sr”
15		“iwanI”			
	R15.1		“jv_sG”		“prv_vN”
	R15.2		“Bv_sG”		“prv_vN”
	R15.3		“sM_bX”		“sM_bX”
16		“liye”			
	R16.1		“apv_sr”	“prv_vN”	“mu_ky”
	R16.2		“upv_sr”	“mu_ky”	“sM_bX”
17		“so”			
	R17.1		“amu_ky”		“vB_smu_bX”
	R17.2		“ap_cn”		“vB_smu_bX”
	R17.3		“mu_ky”		“sMv_sr”
18		“WIM”			
	R18.1		“xv_sG”		“mu_ky”
	R18.2		“amu_ky”		“amu_ky”
19		“hE”			
	R19.1		“mu_ky”		“amu_ky”
	R19.2		“amu_ky”		“amu_ky”
	R19.3			“Bv_sG”	“mu_ky”

i.e. on either side of the considered MCW). As explained in Table 6, Curr_tag: POS tagging of current considered MCW, Prev_tag: POS tagging of immediate previous word to the considered MCW and Next_tag: POS tagging of immediate next word to considered MCW, which helps in generating rules for the list of MCWs' (refer Fig. 9).

The Table 6 consists of six columns, (i) column1 indicates MCW number, numbered as 1,2,3,…,19; (ii) column 2 indicates decision rule number, numbered as R1.1,…,R13.2,…,R19.3; (iii) column 3 indicates MCWs; (iv-v) column 4 and 5 indicate antecedent of the decision rules, which are immediate adjacent (previous and next) POS tags for the corresponding MCW; (vi) column 6 indicates consequent of the decision rules for the corresponding MCW.

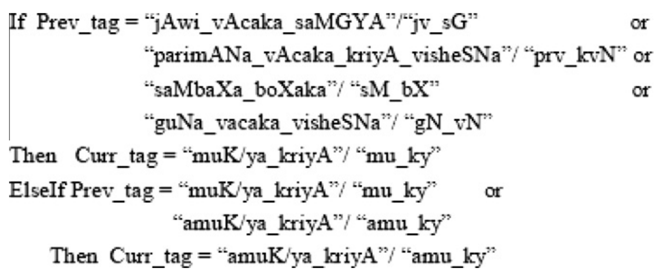
Some instances from Table 6 are discussed below: Consider MCW #1 i.e. “WA” with Rule#1.1 to 1.6. The rules corresponding to “WA” are generated in Fig. 10.

In this case, we do not require Next_tag to generate rules. Only Prev_tag is utilized for the same.

Consider another instance of MCW from Table 6 i.e.MCW#7 “usake” having Rules# 7.1 and 7.2. The rules corresponding to “usake”are generated in Fig. 11.

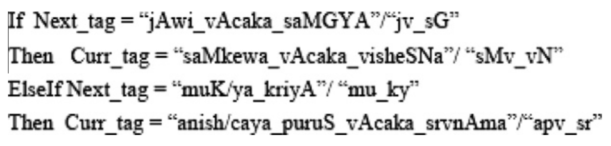
In this case, we do not require Prev_tag to generate rules. Only Next_tag is utilized for the same.

We consider one more instance of MCW from Table 6 i.e. MCW#13 “al\pa_virAma” having Rule# 13.3. The rule corresponding to “al\pa_virAma” is generated in Fig. 12.



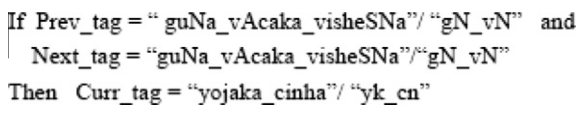
```
If Prev_tag = “jAwi_vAcaka_saMGYA”/“jv_sG”                or
              “parimANa_vAcaka_kriyA_visheSNa”/ “prv_kvN” or
              “saMbaXa_boXaka”/ “sM_bX”                   or
              “guNa_vacaka_visheSNa”/ “gN_vN”
Then  Curr_tag = “muK/ya_kriyA”/ “mu_ky”
ElseIf Prev_tag = “muK/ya_kriyA”/ “mu_ky”     or
                  “amuK/ya_kriyA”/ “amu_ky”
  Then Curr_tag = “amuK/ya_kriyA”/ “amu_ky”
```

Fig. 10. Rules for MCW “WA” within Hindi corpus.



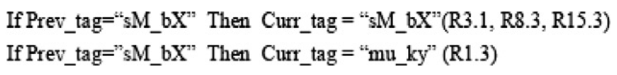
```
If Next_tag = “jAwi_vAcaka_saMGYA”/“jv_sG”
Then  Curr_tag = “saMkewa_vAcaka_visheSNa”/ “sMv_vN”
ElseIf Next_tag = “muK/ya_kriyA”/ “mu_ky”
Then Curr_tag = “anish/caya_puruS_vAcaka_srvnAma”/“apv_sr”
```

Fig. 11. Rules for MCW “usake” within Hindi corpus.



```
If Prev_tag = “ guNa_vAcaka_visheSNa”/ “gN_vN”   and
  Next_tag = “guNa_vAcaka_visheSNa”/“gN_vN”
Then  Curr_tag = “yojaka_cinha”/ “yk_cn”
```

Fig. 12. Rule for MCW “al\pa_virAma” within Hindi corpus.



```
If Prev_tag=“sM_bX”  Then Curr_tag = “sM_bX”(R3.1, R8.3, R15.3)
If Prev_tag=”sM_bX”  Then Curr_tag = “mu_ky” (R1.3)
```

Fig. 13. Sample inconsistent rules within Hindi corpus.

In this case, we require both tags (Prev_tag and Next_tag) to generate rules.

Rest of the rules in Table 6 can be explained on the similar lines.

4.3. Elimination of inconsistent rules in Hindi

Example of an identified inconsistent rules is in Fig. 13.

Fig. 13 shows that rules numbered R3.1, R8.3 and R15.3 are inconsistent with the rule numbered R1.3 (Section 3.2, Step V). Making use of Maximum Likelihood Estimate (MLE) (Konchady, 2009), rules R3.1, R8.3 and R15.3 has maximum likelihood over the rule R1.3 (Table 6). So, eliminating the rule R1.3, rest of the rules R3.1, R8.3 and R15.3 are consistent. Similarly, other inconsistent rules are eliminated from the corpus in this step.

Table A
List of Hindi स्वर“s\vara”, their मात्राएँ “mAtrAeM” and their Hindi transliteration.

S.No.	स्वर	मात्राएँ	Hindi transliteration
1	अ		“a”
2	आ	ा	“A”
3	इ	ि	“e”
4	ई	ी	“E”
5	उ	ु	“u”
6	ऊ	ू	“U”
7	ऋ	ृ	“rh”
8	ए	े	“e”
9	ऐ	ै	“E”
10	ओ	ो	“o”
11	औ	ौ	“O”

In Hindi अनुस्वार “anus\vAra” (ं) and विसर्ग“visarga” (ः) are joined with अ to form अं“aM” and अः“a:” respectively.

Table B
List of व्यंजन “v\yMjana” and their transliteration in Hindi language.

S.No.	व्यंजन “v\yMjana”	Detailed व्यंजन	Devanagari alphabet	Hindi transliteration
1	स्पर्श “s\parSa”			
		कवर्ग “kavarga”		
			क	“k”
			ख	“K”
			ग	“g”
			घ	“G”
			ङ	“dz”
		चवर्ग “cavarga”		
			च	“c”
			छ	“C”
			ज	“j”
			झ	“J”
		टवर्ग “tavarga”		
			ट	“t”
			ठ	“T”
			ड	“d”
			ढ	“Dh”
			ण	“N”
		तवर्ग “wavarga”		
			त	“w”
			थ	“W”
			द	“x”
			ध	“X”
			न	“n”
		पवर्ग “pavarga”		
			प	“p”
			फ	“P”
			ब	“b”
			भ	“B”
			म	“m”
2	अंतःस्थ “aMw:s\W”			
			य	“y”
			र	“r”
			ल	“l”
			व	“v”
3	ऊष्म “US\ma”			
			श	“sh”
			ष	“S”
			स	“s”
			ह	“h”

Some more व्यंजन“v\yMjana” are: श्र“Sra”; क्ष“kS”; ज्ञ“JY” and त्र“ wra” respectively. Also, ढ़“DhZ”; ड़“dZ” are the sounds used in “tavarga” in Hindi.

5. Conclusion

In this paper, we have devised a methodology to identify multi-category words in Hindi language based on the context information and to disambiguate them using rules' based TENGRAM method. Once a consistent set of rules is generated, they can be applied analogusly to any other Hindi corpus so as to remove ambiguity of MCWs'.

We have constructed a Hindi tagger based on Hindi grammar, to deal with different word-order (subject-object-verb) over English grammar order (subject-verb-object). This paper also provides Hindi transliteration which helps non-Hindi readers to pronounce Hindi alphabets into their English character format.

6. Future work

The current work can be extended by performing phrase level POS tagging which can be applied to a group of words at one go. A hybrid approach i.e. stochastic combined with rules based (rough set-hybrid model) can be developed for better results.

Appendix A

Hindi phonology consists of स्वर “s\vara” and व्यंजन “v\yMjana”. We have enclosed Hindi transliterated words within “ ” and they are case sensitive (upper and lower case letters of English language). Also, half – characters are produced using “ \”.

List of Hindi स्वर “s\vara”, their मात्राएँ“mAtrAeM” and their Hindi transliteration is in the Table A. List of व्यंजन “v\yMjana” and their transliteration in Hindi language are in the Table B.

References

Araujo, L. (2002). Part-of-Speech Tagging with Evolutionary Algorithms. In *International Conference on Intelligent Text Processing and Computational Linguistics* (pp. 230–239). Springer-Verlag.

Dalal, A., Nagraj, K., Sawant, U. & Shelke, S. (2006). Hindi Part-of-Speech Tagging and Chunking: A Maximum Entropy Approach. NLPAI Machine Learning Workshop: Part of Speech Tagging and Chunking for Indian Languages.

Demri, S., & Orlowska, E. (2002). *Incomplete Information, Structure, Inference, Complexity*. Heidelberg: Springer-Verlag.

Fahim, H.M., Naushad, U. & Mumit, K.(2007). Comparison of Unigram, Bigram, HMM and Brill's POS Tagging Approaches for Some South Asian Languages. Center for Research on Bangla Language Processing, BARC University.

Konchady, M. (2009). Text Mining Application Programming. Cengage Learning, India Edition.

Lin, T. Y., & Cercone, N. (1997). *Rough Sets and Data Mining- Analysis of Imperfect Data*. Boston: Kluwer Academic Publishers.

Orlowska, E. (1997). *Incomplete Information: Rough Set Analysis*. Heidelberg: Physica-Verlag.

Pawlak, Z. (1982). Rough Sets. *International Journal of Computer and Information Sciences, 11*, 341–356.

Pawlak, Z. (1991). *Rough Sets- Theoretical Aspects of Reasoning About Data*. Boston: Klumer Academic Publisher.

Polkowski, L. (2002). *Rough Sets: Mathematical Foundations*. Heidelberg: Physica-Verlag.

Polkowski, L., & Skowron, A. (1998). *Rough Sets in Knowledge Discovery I and II*. Heidelberg: Physica-Verlag.

Polkowski, L., Tsumoto, S., & Lin, T. Y. (2000). Rough Set Methods and Applications. In *New Developments in Knowledge Discovery in Information Systems*. Heidelberg: Physica-Verlag.

Qian, Yi-li & Zheng, Jia-heng (2004). An Approach to Improving the Quality of Chinese Text. International conference on Information Technology: Coding and Computing. IEEE Computer Society Las Vegas, Nevada, USA, 2, 183-187.

Quinlan, J. R. (1987). Generating Production Rules From Decision Trees. In *Tenth International Joint Conference on Artificial Intelligence* (pp. 304–307). Milan, Italy: Morgan Kaufman.

Rao, D. & Yarowsky, D. (2007). Part of Speech Tagging and Shallow Parsing of Indian Languages. International Joint Conference on Artificial Intelligence Workshop on Shallow Parsing in South Indian Languages, Hyderabad, India.

Ray, P.R., Harish, V., Sarkar, S. & Basu, A. (2003). Part of Speech Tagging and Local Word Grouping Techniques for Natural Language Parsing in Hindi. International Conference on Natural Language Processing, Mysore, India.

Shrivastava, M., Agrawal, N., Mohapatra, B., Singh, S. & Bhattacharya, P. (2005). Morphology Based Natural Language Processing Tools for Indian Languages. Inter Research Institute Student Seminar in Computer Science. India: IIITK.

Sivanandam, S. N., & Deepa, S. N. (2008). *Principles of Soft Computing*. New Delhi, Inida: Wiley India Pvt. Ltd..

Skowron, A., & Rauszer, C. (1992). The Discernibility Matrices and Functions in Information Systems. In *Intelligent Decision Support - Handbook of Applications and Advances of the Rough Sets Theory* (pp. 331–362). Dordrecht: Kluwer Academic Publishers.

Subramanian, G. H., Nosek, J., Raghunathan, S. P., & Kanitkar, S. S. (1992). A Comparison of the Decision Table and Tree. *Communications of the ACM, 35*(1), 89–94.

Tapanainen, P. & Voutilainen, A. (1994). Tagging Accurately – Don't Guess if You Know. In Proceedings of ANLP '94.

Visvendru, R. P. (1993). *Sachitra Hindi Vayakarana Part-I (Class 6)*. New Delhi, India: Saraswati House Pvt. Ltd. Educational Publisher.