A STUDY OF MATHEMATICAL MORPHOLOGY OF GUJARATI SCRIPT USING WAVELET, OPTIMIZATION AND SOFT COMPUTING TECHNIQUES

A Thesis submitted to Gujarat Technological University

for the Award of

Doctor of Philosophy

in

MATHS-SCIENCE

By

MANISHABEN RAMESHBHAI PRAJAPATI

Enrollment No. 149997673007

Under Supervision of

Dr. Archit Yajnik

Co-supervision of

Dr. Trupti P. Shah

GUJARAT TECHNOLOGICAL UNIVERSITY

AHMEDABAD

APRIL-2021

A STUDY OF MATHEMATICAL MORPHOLOGY OF GUJARATI SCRIPT USING WAVELET, OPTIMIZATION AND SOFT COMPUTING TECHNIQUES

A Thesis submitted to Gujarat Technological University

for the Award of

Doctor of Philosophy

in

MATHS-SCIENCE

By

MANISHABEN RAMESHBHAI PRAJAPATI

Enrollment No. 149997673007

Under Supervision of

Dr. Archit Yajnik

Co-supervision of

Dr. Trupti P. Shah

GUJARAT TECHNOLOGICAL UNIVERSITY

AHMEDABAD

APRIL-2021

DECLARATION

I declare that the thesis entitled **"A Study of Mathematical Morphology of Gujarati Script Using Wavelet, Optimization and Soft Computing Techniques"** submitted by me for the degree of **Doctor of Philosophy** in **Maths-Science** is the record of research work carried out by me during the period from December 2014 to April 2021 under the supervision of **Dr. Archit Yajnik, Professor, Sikkim Manipal University,** and this has not formed the basis for the award of any degree, diploma, associateship, fellowship, titles in this or any other University or other institution of higher learning.

I further declare that the material obtained from other sources has been duly acknowledged in the thesis. I shall be solely responsible for any plagiarism or other irregularities, if noticed in the thesis.

Signature of Research Scholar Date:20-4-2021

Name of Research Scholar: **Manishaben Rameshbhai Prajapati**

Place: **Gandhinagar**

CERTIFICATE

I certify that the work incorporated in the thesis **"A Study of Mathematical Morphology of Gujarati Script Using Wavelet Optimization and Soft Computing Techniques"** submitted by **Manishaben Rameshbhai Prajapati** was carried out by the candidate under my guidance. To the best of my knowledge: (i) the candidate has not submitted the same research work to any other institution for any degree, diploma, Associate ship, Fellowship or other similar titles (ii) the thesis submitted is a record of original research work done by the Research Scholar during the period of study under my supervision, and (iii) the thesis represents independent research work on the part of the Research Scholar.

Signature of Supervisor: Date: 20-4-2021

Name of Supervisor: **Dr. Archit Yajnik,**
Professor
Department of Mathematics,
Sikkim Manipal Institute of Technology,
Sikkim Manipal University.
Place: Rangpo, Sikkim

Course-work Completion Certificate

This is to certify that **Ms Manishaben Rameshbhai Prajapati** enrolment no. **149997673007** is a PhD scholar enrolled for PhD program in the branch **MATHS-SCIENCE** of Gujarat Technological University, Ahmedabad

(Please tick the relevant option(s))

☐ He/She has been exempted from the course-work (successfully completed during M.Phil Course)

☐ He/She has been exempted from Research Methodology Course only (successfully completed during M.Phil Course)

☑ She has successfully completed the PhD course work for the partial requirement for the award of PhD Degree. His/ Her performance in the course work is as follows-

Grade Obtained in Research Methodology (PH001)	Grade Obtained in Self Study Course (Core Subject) (PH002)
BB	**BC**

Supervisor's Sign

Dr. Archit Yajnik

Name of Supervisor

Originality Report Certificate

It is certified that PhD Thesis titled entitled **"A Study of Mathematical Morphology of Gujarati Script Using Wavelet, Optimization and Soft Computing Techniques" by** Manishaben Rameshbhai Prajapati has been examined by us. We undertake the following:

a) Thesis has significant new work / knowledge as compared already published or are under consideration to be published elsewhere. No sentence, equation, diagram, table, paragraph or section has been copied verbatim from previous work unless it is placed under quotation marks and duly referenced.
b) The work presented is original and own work of the author (i.e. there is no plagiarism). No ideas, processes, results or words of others have been presented as Author own work.
c) There is no fabrication of data or results which have been compiled / analyzed.
d) There is no falsification by manipulating research materials, equipment or processes, or changing or omitting data or results such that the research is not accurately represented in the research record.
e) The thesis has been checked using Urkund anti plagiarism system (copy of originality report attached) and found within limits as per GTU Plagiarism Policy and instructions issued from time to time (i.e. permitted similarity index <10%).

Signature of the Research Scholar: Date: 20-4-2021

Name of the Research Scholar: Manishaben Rameshbhai Prajapati

Place: Signature of the Supervisor: Date:20-4-2021

Name of Supervisor: Dr. Archit Yajnik

Place: Rangpo, Sikkim

Curiginal

Document Information

Analyzed document	Thesis_copy.docx (D100203625)
Submitted	3/30/2021 4:14:00 PM
Submitted by	Manisha
Submitter email	purvansi263@gmail.com
Similarity	6%
Analysis address	purvansi263.gtuni@analysis.urkund.com

Sources included in the report

SA	**Gujarat Technological University / Final_check.docx** Document Final_check.docx (D100153811) Submitted by: 159997119008@gtu.edu.in Receiver: 159997119008.gtuni@analysis.urkund.com	35
W	URL: https://www.iiitmk.ac.in/vrclc/wp-content/uploads/2018/02/Part-Of-Speech-Tagging-F ... Fetched: 10/31/2019 2:20:41 AM	4
W	URL: http://web2py.iiit.ac.in/research_centres/publications/download/mastersthesis.pdf. ... Fetched: 12/14/2020 8:54:17 AM	1
W	URL: https://scholar.colorado.edu/downloads/2f75r821k Fetched: 6/4/2020 8:07:54 AM	1
SA	**Gujarat Technological University / Final_thesis_1.docx** Document Final_thesis_1.docx (D100201729) Submitted by: purvansi263@gmail.com Receiver: purvansi263.gtuni@analysis.urkund.com	2
SA	**Part of Speech Tagger.docx** Document Part of Speech Tagger.docx (D79008426)	2
SA	**file.docx** Document file.docx (D21530612)	1

PhD THESIS Non-Exclusive License to

GUJARAT TECHNOLOGICAL UNIVERSITY

In consideration of being a PhD Research Scholar at GTU and in the interests of the facilitation of research at GTU and elsewhere, I, **Manishaben Rameshbhai Prajapati**, having Enrollment No. **149997673007** hereby grant a non-exclusive, royalty free and perpetual licence to GTU on the following terms:

a. GTU is permitted to archive, reproduce and distribute my thesis, in whole or in part, and / or my abstract, in whole or in part (referred to collectively as the "Work") anywhere in the world, for non-commercial purposes, in all forms of media;

b. GTU is permitted to authorize, sub-lease, sub-contract or procure any of the acts mentioned in paragraph (a);

c. GTU is authorized to submit the Work at any National / International Library, under the authority of their "Thesis Non-Exclusive Licence";

d. The University Copyright Notice © shall appear on all copies made under the authority of this license;

e. I undertake to submit my thesis, through my University, to any Library and Archives. Any abstract submitted with the thesis will be considered to form part of the thesis.

f. I represent that my thesis is my original work, does not infringe any rights of others, including privacy rights, and that I have the right to make the grant conferred by this nonexclusive license.

g. If third party copyrighted material was included in my thesis for which, under the terms of the Copyright Act, written permission from the copyright owners is required, I have obtained such permission from the copyright owners to do the acts mentioned in paragraph (a) above for the full term of copyright protection.

h. I retain copyright ownership and moral rights in my thesis, and may deal with the copyright in my thesis, in any way consistent with rights granted by me to my University in this non-exclusive license.

i. I further promise to inform any person to whom I may hereafter assign or license my copyright in my thesis of the rights granted by me to my University in this non-exclusive license.

j. I am aware of and agree to accept the conditions and regulations of PhD including all policy matters related to authorship and plagiarism.

Signature of Research Scholar

Name of Research Scholar: **Manishaben Rameshbhai Prajapati**

Date: 20-4-2021

Place: Gandhinagar

Signature of Supervisor

Name of Supervisor: Dr. Archit Yajnik

Date:20-4-2021

Place: Rangpo, Sikkim

THESIS APPROVAL FORM

The viva-voce of the PhD thesis submitted by **Ms Manishaben Rameshbhai Prajapati**, enrollment no. **149997673007** entitled "**A Study of Mathematical Morphology of Gujarati Script Using Wavelet, Optimization and Soft Computing Techniques**" was conducted on 20-04-2021 at Gujarat Technological University.

Please tick any one of the following options

☑ The performance of the candidate was satisfactory. We recommend that he should be awarded the Ph.D. degree.

☐ Any further modifications in research work recommended by the panel after 3 months, from the date of first viva-voce, upon request of the supervisor or request of independent research scholar after which, viva-voce can be re-conducted by the same panel again.

☐ The performance of the candidate was unsatisfactory. We recommend that he/she should not be awarded the Ph.D. degree.

Dr. Archit Yajnik, Supervisor

Dr. Trupti P. Shah, Co- Supervisor

Signature of Examiner
Name of Examiner: Dr. Raju K. George
Place:

Dr. Raju K George, External Examiner-1

Dr. Vikas H. Pradhan, External Examiner-2

ABSTRACT

The thesis presents the various Part of Speech (POS) tagging technique and Parsing for Gujarati Languages. Both the techniques are play a vital role in the development of the Natural Language Processing (NLP) based tools like Morphological analyser, Machine translation etc.

Many algorithms have been developed for part of speech (POS) tagging. Support Vector Machine (SVM), Hidden Markov Model (HMM), Maximum Entropy Markov Model (MEMM), Neural Networks (NN), Decision Trees, Rule based and Transformation based techniques are to name a few. Following are research outcomes:

1. Generate the annotated corpus for Gujarati language
2. Viterbi based POS tagger
3. SVM based POS tagger
4. Linear Programming Problem (LPP) based Parser
5. New Technology is developed on conceptual graph to parse the Projective Sentences.

The thesis is about performing Part of speech tagging to the Indian language Gujarati using Hidden Markov Model and SVM Tool and its Parsing. The Hidden Markov Model gives the 92% accuracy. The SVM Tool is giving accuracy of 72%. In Parsing we use New terminology Name Conceptual Graph.

This thesis will give an idea about various algorithms used for part of speech tagging followed by SVM and Viterbi. It presents a clear idea about all the algorithms from the ground level with suitable examples. It also introduces a tag set for Guajarati which can be used for tagging Gujarati text. A tool which helps in tagging the text is also introduced. After tagging, the outputs were also compared to check the accuracy.

Natural Language Processing (NLP) is a challenging field in the area of artificial intelligence and computational linguistics. In simplistic terms, it can be defined as processing a natural

language in any form either speech or written text. Though extensive research has been done for many of the languages in the world, Indian languages are still lagging behind in the race.

Processing of any natural language requires analysis to be done at multiple levels like word-level, phrase-level, sentence-level, semantic-level and higher levels of pragmatic and discourse. In this work we are presenting our efforts of making new advancements at the sentence level, which in linguistics terms, is regarded as Syntactic Parsing. Syntactic parsing involves establishing relations between different words of a sentence to convey the possible meaning.

Indian languages are morphologically rich, free word order language. Parsing morphologically rich, free word order languages (MoR-FWO) is a challenging task. In this work we present our experiments which lead to state-of-the-art dependency parser for Gujarati. We do a series of experiments exploring the role of different morphological and syntactic features in Gujarati dependency parsing using Grammar Driven Parsing and Conceptual Graph. During the course of experiments, we realized that some of the basic linguistic constraints were violated by these data driven parsers. We built a linguistically sound parser without compromising on the accuracy. Consider a simple linguistic constraint that a verb should not have multiple karta karakas (roughly, subjects) as its children in the dependency tree. We propose two approaches to handle this constraint and evaluate them on the state-of-the-art Gujarati dependency parser.

Acknowledgement

This thesis is truly dedicated to HDH **Pramukh Swami Maharaj** without blessing of him I cannot able to complete it. I dedicate this thesis for tribute on the occasion of Pramukh Swami Maharaj Shatabdi Mahotsav. I dedicate thesis to the Charnavind of **Purna Purushottam Bhagvan Swaminarayan** and **Aksharbrahman Gunatitanand Swami** and **HH Mahant Swami Maharaj**.

Firstly, I would like to express my sincere gratitude to my advisor **Prof. Archit Yajnik** for the continuous support of my Ph.D. study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D. study. He always supported intellectual freedom, permitted my attendance at various conferences, and always demanded high quality of work.

While the Gujarat Technological University, Ahmedabad has been supportive enough for conduction of half yearly Doctoral Progress Committee (DPC) reviews, besides my advisor, I would like to thank the rest of my thesis committee: Co-Supervisor Dr. Trupti P Shah, Dr. Jayesh M. Dhodiya, and Dr. S. Rammohan for their insightful comments and encouragement, but also for the hard question which incanted me to widen my research from various perspectives.

Foremost, I would like to thank Dr. Abdul Zummervala for providing technical assistance for the data processing. Friendly and cooperative atmosphere at work and also useful feedback, could not have been possible without my friends.

Last but not the least, I would like to thank my Father Rameshbhai, Mother Bhartiben, Soulmate Apurva and my loving daughter Purvanshi. Brother Shrikant, Sister Pratiksha and my two-little champ Pranshi and Yashika for their love and understanding that allowed me to pursue higher education and devote my time to achieve something of significance.

Manisha Prajapati

Table of Content

List of Abbreviations

Abbreviation	Full form
NLP	Natural Language processing
IVR	Interactive Voice Response
PC	Personal Computer
NER	Named entity recognition
POS	Part-of-speech Tagging
MLE	maximum-likelihood estimation
HMM	Hidden Markov Model
SVM	Support Vector Machine
CRF	Condition Random Field
MST	Maximum Spanning Tree
MOR-FWO	Morphological free word order
CFG	Context Free Grammar
LAS	Labeled Attachment Score
NL	Natural language
PDT	Prague Dependency Treebank
DDT	Danish Dependency Treebank
TAM	Tense aspect modality
LPP	Linear Programming Problem
BG	Bipartite Graph
OBG	Order Bipartite Graph
CG	Conceptual Graph
NL-Support	Natural Language Support
NL-Labelling	Natural Language Labelling
ANN	Artificial Neural Network
DWT	Discrete wavelet transform
CWT	Continuous wavelet transform
SVD	Singular Value Decomposition
MT	Machine Translation
ATIS	Automatic Terminal Information Service
IO	Indirect Object
DO	Direct Object

List of Symbols

No	Symbol	Meaning
1	≈	Approximately equal to
2	∪	Union
3	∩	Intersection
4	⊂	Proper Subset of
5	←	snippet from some pseudo code for a sorting algorithm, used to denote assignment, for example for the variable done
6	→	Edge from A to B
7	↔	Edges from both side
8	$d_G(r)$	degree of r in G
9	∅	Empty set
10	$L^2(R)$	Set of Square integrable real valued functions. $L^2(R) = \int_{-\infty}^{\infty} \lvert f(x)\rvert^2\, dx$
11	Span	The span of subspace generated by vectors $v_1 and\ v_2 \in V$, Span $(v_1, v_2) \equiv \{rv_1 + sv_2\ r, s \in R\}$
12	≡	Identical to
13	⊆	Is a subset of
14	∨(OR)	Logical Disjunction $\{A \vee B$ is true iff A or B (or both) are true.
15	∧(And)	Logical Conjunction. {A$\wedge B$ is true iffA is true and B is true.}
16	¬	The statements $\neg A$ is true iff A is False.

List of Figures

List of Tables

List of Appendices

CHAPTER-1

Introduction

Natural Language Processing (NLP) is the technology which is used to help computers to understand the human's natural language. It is a branch of artificial intelligence that manages the connection between computers and humans using the natural language. Most NLP techniques rely on machine learning to derive meaning from human languages. The aim of a linguistics science is to be able to characterise and explain the multitude of linguistics observation float around us, in discussions, composing, and other media. Part of that has to do with the intellectual size of how human acquired, produce and understand language, part of it has to do with understanding the relationship between linguistics utterness and the world, and part of it has to do with understand the linguistics structures by which language communicates. NLP is the driving force behind the following common applications such as Google Translate, Word Processors and Personal assistant applications like OK Google, Siri, Cortana, and Alexa. Microsoft Word and Grammarly utilize NLP to check grammatical accuracy of text. Interactive Voice Response (IVR) applications utilized in call centres to respond to specific clients' requests. NLP is considered a complex problem in computer science. It's the nature of the human language that makes Natural language processing troublesome. The principles that identify the passing of information using natural languages aren't easy for computers to understand. A portion of these guidelines can be high-levelled and unique for instance when somebody utilizes a mocking comment to pass data. On the other hand, some of these rules can be low-levelled for example, using the character "s" to mean the plurality of objects. Exhaustively understanding the human language requires grasping both the words and how the ideas are associated with convey the expected message. While people can easily master a

language, the ambiguity and lose characteristics of the natural languages what make NLP hard for machines to implement. NLP requires applying algorithms to identify and remove the natural language rules such that the unstructured language data is transformed in a way that computers can understand. At the point when the content has been given, the computer will do calculations to extract significance related to each sentence and gather the basic information from them. Sometimes, the computer may fail to understand the meaning of a sentence well, leading to doubtful results. For example, the translation of a few words between the English and the Gujarati languages.

E.g. The following sentence requires translation:

"Marry is Good in nature".

The result, when the sentence has been translated from English to Gujarati is:

"લગ્ન પ્રકૃતિ સારી છે."

It is a vague translation that does not have any context as far as the Gujarati is a concern. Syntactic analysis and semantic analysis are the most techniques to be complete Natural Language Processing tasks. The explanation of the notions used in NLP is as follows. Here may be a description of how they will be used.

1.1 Syntax:

It refers to the arrangement of words in during a sentence in such a way that they create grammatical sense. In NLP, syntactic analysis is employed to judge the linguistic communication with the grammatical rules. Computer algorithms apply grammatical rules to a group of words and derive meaning from them. The syntax methods which are commonly used is as follows:

- Lemmatization: It involves reducing the varied inflected forms of a word into single form for simple analysis [65].
- Word segmentation: It involves dividing a large piece of continuous text into distinct units.
- Part of Speech tagging: It involves identifying the part of speech for each word.

- Parsing: It involves undertaking grammatical analysis for the given sentence.
- Sentence breaking: It involves placing sentence boundaries on an large piece of text.
- Stemming: It involves cutting the inflected words to their root form.

1.2 Semantics:

Semantics refers to the meaning that's conveyed by a text. Semantic analysis is one of the difficult aspects of NLP that has not been fully resolved yet. It includes applying computer algorithms to grasp the meaning and interpretation of words and how sentences are structured. Here are some methods used in semantic analysis:

- Named entity recognition (NER): It includes deciding the pieces of a text which will be recognized and ordered into preset groups. Samples of such groups include names of people and places.
- Word meaning disambiguation: It includes offering significance to a word supported the precise situation.

1.3 Linguistic to NLP:

1.3.1. Linguistic:

It is the scientific study of language including its grammar, semantics, and phonetics. Classical linguistics involved devising and evaluating rules of language. Great progress has been made on formal methods for syntax and semantics. Except for the foremost part, the interesting problems in linguistic communication grasping resist clean mathematical formalisms. Broadly, a linguist is anyone who studies language, but perhaps more informal, a self-defining linguist could also be more focused on being move into the sector. Mathematics is the tool of science. Mathematicians working on linguistic communication refer to their study as mathematical linguistics, focusing exclusively on the utilization of discrete mathematical formalisms and theory for linguistic communication (Formal Language).

1.3.2 Computational Linguistics:

It is the trendy learning of linguistics using the tools of computer science. Yesterday's linguistics could also be today's computational linguistic because the use of computational tools and thinking has overtaken most fields of study. Linguistics is the study of computer systems for understanding and generating natural language. One natural function for linguistics would be the experiment of grammars proposed by theoretical linguists. Within the year 1990, statistical methods and statistical machine learning began to and eventually replaced the classical top-down rule-based approaches to languages, primarily due to their better results, speed, and hardiness. The statistical approach to studying natural language now dominates the sector; it's going to define the sphere. Data-driven methods for natural language processing have now become so popular that they need to be considered mainstream approaches to linguistics. A strong contributing factor to the current development is no doubt the rising amount of accessible electronically stored data to which these methods are often applied, but another factor could be a particular dissatisfaction with approaches which depend on exclusively available made rules, because of their notice brittleness.

1.3.3 Statistical NLP:

Computational linguistics also become known by the name of natural language procedure or NLP, to mirror the more architect based or exact methodology of the statistical method. The statistical dominance of the field additionally frequently prompts NLP to be portrayed as Statistical Natural Language Processing, perhaps to isolate it from the traditional computational linguistics method. The view of computational linguistics as having both a scientific and an engineering side. The engineering side of computational linguistics regularly called natural language processing (NLP) is to a great extent worried about structure computational tools that do valuable things with language. For example, machine translation, summarization, question-answering, etc. Like any engineering branch natural language processing draws on a variety of different scientific order.

Linguistics is a huge subject of study and even though the measurable way to deal with NLP has demonstrated incredible achievement in certain territories but there is still room by the old-

style top-down strategies. Generally, factual NLP partner's probabilities with the choices experienced for breaking down an articulation or a book and acknowledges the most possible result as the right one. Predictable words that name phenomena that are closely related within the world, or our perception of it, frequently occur on the brink of each other so that crisp facts about the planet are reflected in somewhat fuzzier facts about texts. There's much room for debate during this view.

1.4 Part-of-Speech -Tagging (POS - Tagging):

The first step in natural language processing involves the technique of morphology. It involves defining the functions of individual words and their relationships with other words in the same sentence. Most people will be familiar with the simplified form of the process from school, where we are taught that words can be defined either as nouns, verbs, adverbs or adjectives. But determining of word's function for a computer is not a such a simple task. There are various methods to assist an attempt to map out the ambiguity of words with multiple functions and meanings: The oldest and most classical method is predicated on the various text corpora like the Brown corpus or British National corpus. These corpora consist of an millions of words from prose texts that are tagged. Computers can learn rules for the way that different words are tagged in these texts. As an example, the Brown corpus has consistently been utilized to prove to computers that a verb no longer has predicate function if there is an article before it. The present tagging programs utilize self-learning algorithms. This means that they automatically derive rules from the text corpora as they read, using these to define further word function. One among the best-known samples of a tagging method supported algorithms is the Brill tagger. A rule might be something like this: 'If the first word of the sentence may be a proper noun, then the second word is probably going to be a verb'. This is often a standard theory, so within the sentence 'Jay bought a book', the word 'bought' is often defined as a verb.

1.4.1 Classification:

There are different approaches for POS Tagging. The following figure classifies different POS Tagging models. Here we give a very brief overview of the different models [42,63,101].

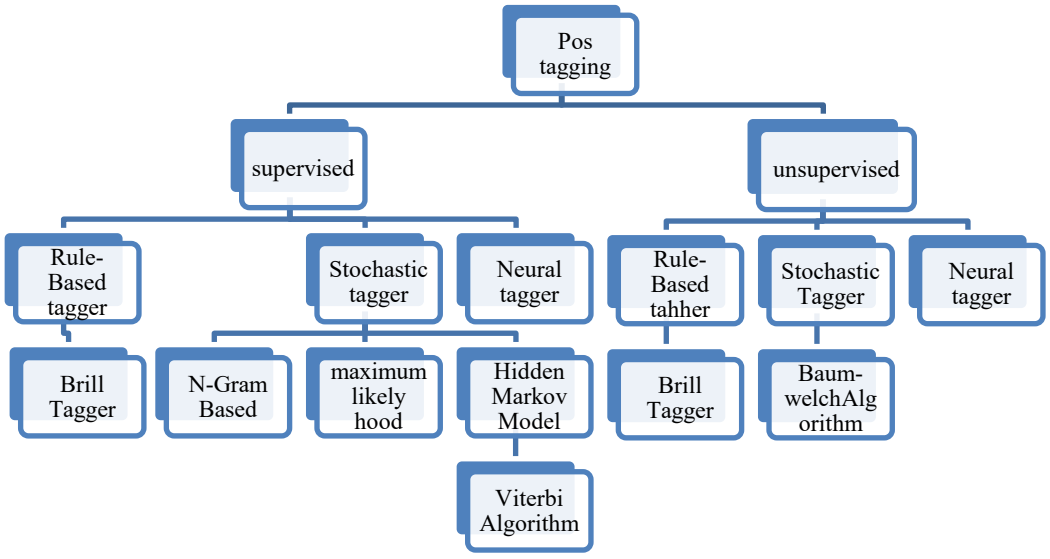


FIGURE 1.1 Classification of POS tagging

1.4.1.1 Supervised models:

The supervised POS Tagging models require a pre-taged corpus which is used for training to learn information about the tagset, word-tag frequencies, rule sets etc. The presentation of the models generally increases with an increase in the size of the corpus.

1.4.1.2 Unsupervised models:

The unsupervised POS Tagging models don't require a pre-annotated corpus. Instead, they utilize advanced computational technique just like the Baum Welch algorithm to consequently prompt tagset, transformation rules etc. Supported with this information they either calculate the probabilistic information needed by the stochastic taggers or induce the contextual rules needed by rule-based systems or transformation-based systems.

Both the supervised and unsupervised models can be further classified in to the following categories

1.4.1.3 Rule-Based Tagger:

Rule-Based taggers generally apply a large data based on handwritten disambiguation rules that specify, for example, that an ambiguous word may be a noun instead of a verb if it follows a determiner. One among the foremost comprehensive rule-based approach is that the constraint grammar approach.

1.4.1.4 Transformation-Based Tagger (Brill Tagger):

The transformation-based approaches use a predefined set of handcrafted rules and also automatically-induced rules that are generated during training. It Shares feature of both tagging architecture, like the rule-based tagger here its supported rules determine when an ambiguous word should have a given tag and like the stochastic tagger, which is a machine learning component, the principles are automatically induced from a previously tagged training corpus.

1.4.1.5. Stochastic Tagger:

The stochastic models include frequency, probability or statistics. They supports different methods like n-grams, maximum-likelihood estimation (MLE) or Hidden Markov Models (HMM). HMM, based technique requires evaluation of the arg max formula, which is over the highest expensive as all possible tag sequences must be checked, In order to find the sequence that maximise the probability. So a dynamic programming approach referred to as the Viterbi Algorithm is used to find out the optimal tag sequence. There have also been several studies utilizing unsupervised learning for training a Hidden Markov Model for Part of speech Tagging. The foremost widely known is that the Baum-Welch algorithm, which may be use to train a HMM from un-annotated data. And lastly, both supervised and unsupervised POS Tagging models are can be based on neural networks.

1.5 Tag-set:

A tagset consist of tags that are used to represent the grammatical information of the language. The number of tags that use for a language depends upon the information that would like to represent using a tag. A tagset is often overlarge according the requirement of the researcher. For representing the context of words during analysis of sentence, various tags are used. For example, if a word is acting as a noun, then (NN) tag is employed, for Pronoun (PRP) tag, Verb(V), Adjective (JJ), Conjunction (CC) are often used [68,75]. Standard in Indian languages (LDC-IL) tagset used for the Gujarati Language.

1.6 Morphology:

The analysis of morphology is isolated into two primary fields as inflection and derivation. Consequently, the morphological structure of every word may include element such as prefix, suffix, infix, or maybe a separate root. These components can adjust the importance of the elemental root or harsh of the word. Within the event that the next word is simply a paradigmatic utilization of its base structure, this type of the word is called Inflection. however, within the event that the next word is an altogether extraordinary word or a compound, which is framed of a minimum of two roots, it's called Derivation. The input verb, which precedes the suffixes, is analyzed as an invariant root by querying the database, and so the subsequent suffix particles may indicate voice (causative, reciprocal, reflexive, passive), modality (necessities, abilitative, conditional), negation, tense-aspect mood, and person/number. Indian Languages are morphologically rich, free ordering languages. As free ordering languages are often handled better using the dependency-based framework. Dependency annotation using the Paninian framework is started for Indian Languages.

1.7 Chunking in NLP:

When we memorize a phone number or the other sequence of numbers, we don't tend to memorize them as separate individual numbers, we group them to form them easier to recollect, this is often called chunking. Chunking is that, the together of information [59].

1.8 Parsing:

Knowledge derived from syntax is used to understand the structure of sentences. Here, the computer linguistics program utilizes tree diagrams to separate a sentence into phrases. Example of phrases are nominal phrases, consisting of proper noun or noun and an article or verbal phrases, which consists of a verb and a nominal phrase. Separating a sentence into phrases is known as 'parsing' thus the tree diagram that result from it, are known as parse trees. For example, Every language has its own grammar rules, meaning that phrases are put together differently in all of which the hierarchy of various phrases vary. Grammar rules for a given language are often programmed into a computer programme by hand, or learned by using a text corpus to recognise and understand sentence structure.

રામે રાવણ ને તિર થી માર્યો. (Ram Killed Ravan with arrows) (Appendix III)

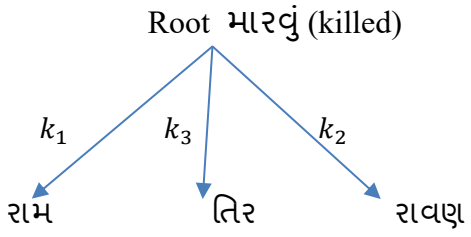


FIGURE 1.2: Parsing of the sentence

1.9 Support Vector Machine:

SVM employs kernel tricks and maximal margin concepts to perform better in non-linear and high-dimensional tasks. Even a strong SVM model, most of the days, have the benefit of the right feature selection and feature extraction/transformation techniques. Artificial Neural Network concept wasn't something new to the computer science world. One-layer perceptron in

artificial neural network didn't perform up to the expectations because it could only capture limited linear patterns. Stacking two or more neural layers i.e., feed-forward neural net or multilayer perceptron improved the performance, but still cannot predict an XOR function.

1.10 Current status of Gujarati Language:

A Machine learning algorithm for Gujarati in part of Speech Tagging has been utilized by Chirag Patel and Karthik Gali [64]. The machine learning part is performed by employing a CRF model. With training corpus is of 10,000 words and the testing corpus of 5,000 words, the algorithm gives 92% accuracy for Gujarati texts.

Sirajuddin Y. Hala and, Sagar H. Virani [103] improved accuracy of Parts of Speech tagger for Gujarati language by using the hybrid approach. They employed both stochastic and rule-based models for tagging. It'll resolve ambiguity and incorrect assigned tags to a word by applying the Linguistic rules of Gujarati. On completion of the system implementation– precision, recall, and measure are going to be calculated and compared with the existing system. Supported that improvement within the proposed system is going to be done.

Mr.Vipul Gamit, Rutva Joshi and Ekta Patel nicely reviewed on Part-Of-Speech Tagging on Gujarati Language [101]. They include the classification of various methods used for parts of speech tagging in the Gujarat language. Rule-based method uses linguistic rules to assign the right tags to the words within the sentence or file. The Stochastically approach finds out the foremost frequently used tag for a selected word within the annotated training data and uses this information to tag that word within the annotated text. In hybrid method, words during this technique are first tagged probabilistically then as post-processing, linguistic rules are applied to tag tokens. We also use these methods to tag the words given within the particular sentence.

Chaudhari Shailesh A. and Gulati Ravi M. defies the words extraction from printed bilingual Gujarati-Roman script documents [27]. They have developed different methods for calculating the dimensions of the structuring element of morphological dilation, combined with different forms and tested on samples of printed Gujarati and Roman documents. Then, they compared

the results, and the best performing was chosen for testing printed bilingual documents in their our study. The proposed method has been applied to the segmentation of Gujarati and Roman documents and proven to be effective.

1.11 Key contribution of the thesis:

The following tools and technologies are being used to carry out the research.

- Part of Speech tagging techniques:
  - Hidden Markov model (Viterbi algorithm)
  - Support Vector Machine
  - Artificial Neural Network-based technique
- Dependency parsing techniques:
  - Insight into Non-projectivity in Gujarati
  - Bipartite Graph and maximum matching graphs
  - Maximum Spanning Tree
  - Linear Programming Problem
  - Conceptual Graph-Based Parsing.
- Identification of Projective sentences using planar graph.

Hidden Markov model with Viterbi algorithm gives 92.87% accuracy. The matlab code with 1500 sentences used to derive this result. With the same no. of 1500 sentences as a training data, we were able to build a SVM using Python Programming which gives accuracy of 74.5%. The major contribution of this thesis is the state-of-the-art dependency parser for Gujarati. We used grammar-driven parsers like integer programming, maximum matching, Bipartite Graph, and Conceptual Graph. We did a series of experiments exploring the role of different morphological and syntactic features in Gujarati dependency parsing. The goal of this work is to build a linguistically sound parser without compromising on the accuracy. A step by step analysis of the different features like Part-Of-Speech, morph and chunk information for sentence-level parsing of Gujarati is done.

1.12 Outline:

Chapter 2: This chapter covers a Literature survey of the parsing and NLP. It presents a review of the state work till date which naturally lead to the present topic of research.

Chapter 3: This chapter covers the Markov chain and discusses the statistical information about the Hidden Markov Model. Detailed discussion about the Emission Probability and Transition Probability is also carried out. The Viterbi algorithm and its functioning are also covered.

Chapter 4: This chapter describes the support vector machine and feature extraction technique. It also discusses the implementation of a multi-class problem, its error analysis and about the wavelet-based neural network.

Chapter 5: This chapter focuses on the cases of non-projectivity for Guajarati sentences.

Chapter 6: In this chapter, description of a Paninian Grammar for Gujarati language is given. Detailed study of the Karaka Relation for Gujarati language, Karaka to Vibhakti mapping, TAM labeled and Verb Frame for Gujarati Language is made. It also focuses on the importance of basic linguistic constraints in statistical dependency parsing, need of Parsing and types of Parsing. The different technique of Parsing for Gujarati Language like Parsing using Bipartite Graph, Maximum Matching etc. are discussed in this chapter. It also discusses about the Planar Graph and Parsing using LPP. We apply these approaches to handle the state-of-the-art dependency parsers for Gujarati.

Chapter 7: This Chapter discusses about Conceptual Graphs. It also discussing use of Ordered Bipartite Graph, NL- Support and Labelling in the Gujarati Parsing.

Chapter 8: The last chapter is about the conclusion of the thesis and some future scope of the research done so far.

CHAPTER-2

Literature Survey

As compare to foreign languages, a really little work has been wiped out the natural language processing for Indian languages. Various Parsers for Indian Languages like Marathi, Kannada, Bengali, Hindi and Assamese and Telugu are available but it's still an ongoing process for Indian languages.

To improve Malt Parser for three Indian languages Hindi, Bangla and Telegu in NLP Tools Contest at ICON 2009. They accomplished second position among took part frameworks. A little test set of 150 sentences was utilized to examine the exhibition of the framework. The presentation of the framework was marginally better for Bangla and Hindi language however for Telugu it was lower than the pattern results. The idea was given by Joakim Nivre in 2009 [56].

The developed parser uses a bidirectional parsing algorithm with two operations projection and non -projection to create the dependency tree for the Hindi, Telugu and Bangla languages. He gives a labelled attachment score of 59.86%. 67.74% and 71.63% for Telugu, Bangla and Hindi respectively on the Treebank with fine-grained dependency labels. It had been proposed by Prashanth Mannem in 2009[1].

Three Indian languages namely Hindi, Bangla and Telugu for Data-driven parsers called Malt and MST. They merged both the training and development data and did 5-fold cross-validation best

settings from the cross-validation experiments and these settings are applied on the test data of the competition They found that for all the languages Malt performed better over Maximum Spanning tree and maximum entropy. They reported that, the common of best unlabelled attachment gives 88.43%, labelled attachment gives 71.71% and labelled accuracy are 73.81%. The implementation of MST to handle vibhakti and TAM markers for labelling. This analysis explored by Bharat Ram Ambati et al. in 2009 [18].

Akshar Bharati et al. in 2009 [21] describe syntactic parser which is follows by grammar driven methodology. It provides a dependency grammar framework which is on Paninian grammatical approach, a proposed parser.

A constraint based Hindi dependency parser. In the proposed framework a sentence structure driven methodology was supplemented by a controlled measurable system to accomplish superior and heartiness. The created framework utilizes two phase limitation based half breed way to deal with reliance parsing. From the experiment they acknowledged that the simplest labelled attachment score is 62.20% and unlabelled attachment accuracy are 85.55% for Hindi. Which was proposed by Kalyan Deepak and Meher Vijay Yeleti in 2009 [53].

A data driven dependency parsing approach which utilizes data about the provisos in a sentence to improve the performance of a parser. They demonstrated a MOR-FWO language experiments on Hindi, they used a improved version of MST Parser. They achieved accuracy of 0.77% in labelled attachment and 0.87% in unlabelled attachment. It was described by Phani Gadde et al. in 2010 [40].

The parallel importance of various linguistic features for data-driven dependency parsing of Hindi. They had joined the features from the two different parsers and achieved a labelled attachment score of 76.5%, which is 2 % points better than the previous state of the art. It was analysed by Bharat Ram Ambati et al. in 2010 [3].

Lexicon Parser for Devanagari script (Hindi) shows that how a sentence is parsed into tokens. They used Rule based approach to resolves the disambiguity of words, then find the connection between tokens using grammar and semantic representation generate a parse tree. The accuracy was 89.33% achieved by Lexicon parser. From the research, it has been seen that the exactness was low when they tried greater ambiguity sentences and sentences of future tense. Thus, when they tried sentences of simple present and past tenses then the accuracy was high. It was developed by Swati Ramteke et al in 2014 [95].

A constraint-based Dependency parsing., a free-word order Language for Bangali. The fundamental idea behind this approach is to simplify the complex and compound sentential structures. A well-known and extremely effective grammar formalism for free of charge ordering language called Paninian Grammatical model was used for this purpose. Then to parse the easy structures so obtained by satisfying the Karaka demands of the Verb Groups and to re-join such parsed tree with suitable links and Karaka tags. The performance of the system was evaluated with 150 sentences and accuracies achieved are of 79.81%, 90.32%, and 81.27% and for labelled attachments, unlabelled attachments and label scores respectively. It proposed by Sankar De et al. in 2009 [82].

Statistical Condition Random Field based model followed by a rule-based post-processing technique has been used for Bengali. The results an unlabelled attachment score of 74.09%, labelled attachment score of 53.90% and labelled accuracy score of 61.71% respectively. The system demonstrated by Aniruddha Ghosh et al. in 2009 [5]. The baseline Condition Random Field based system is filtered by a rule-based post-processing module by using the output obtained through the rule-based dependency parser.

A hybrid approach for parsing Bengali sentences. The proposed system was supported data driven dependency parser. It had been research by Sanjay Chatterjee and et al. in 2009 [81].

A grammar formalism called the ‘Paninian Grammar Framework’ that has been successfully applied to all or any free word Indian languages. They have described a constraint-based parser. It's found that the Paninian framework applied to modern Indian languages will provides a chic account of the relation between vibhakti and karaka roles which the mapping is elegant and compact. It had been described by Akshar Bharati and Rajeev Sangal [22].

The way of manufacturing context free grammar for the noun phrase and predicate agreement in Kannada Sentences. The system works in two levels: First of all, it generates the CFG of the sentence. Within the second level, a recursive descent parser called Recursive Descent Parser of natural language tool kit (NLTK) was wont to test the grammar. They need tested the system with 200 sample sentences and obtained encouraging results. It was proposed by B.M. Sagar et al in 2009 [15].

The famous grammar formalism called Penn Tree bank structure was wont to create the corpus for proposed statistical syntactic parser for Kannada language. The evolve corpus has been already annotated with correct segmentation and Part-Of-Speech information. The developers used their own Support vector machine based a part of speech tagger generator for assigning proper tags to every and each word within the testing and training sentences. Training, testing and evaluation were done by support vector method algorithms. Experimental observations show that the performance of the proposed system is significantly good and has very competitive accuracy. It had been developed by Antony P J et al. in 2010 [44].

A Context Free Grammar (CFG) analysis for easy Kannada sentences. They need explained the writing of Context Free Grammar (CFG) for an easy Kannada sentence with two types of examples. Within the developed system, a language grammar is parsed with Top-Down and Bottom-Up parsers and that they found that a Top-Down parser is more suitable to parse the given grammatical production. It was proposed by B.M. Sagar et al. in 2010 [77].

A context free grammar for easy Assamese sentences. They need analysed the problems that arise in parsing Assamese sentences and produce an algorithm to unravel those issues. The algorithm may be a modification of Earley's Algorithm and that they found the algorithm simple and efficient. It was developed by Rahman, Mirzanur and et al. in 2009 [72].

A parsing criterion for Assamese text described by Navanath Saharia et al. in 2011 [78]. This approach will be to parse the straightforward sentences with multiple noun, adjective, adverb clauses.

To write down context free grammar for easy Marathi sentences. Grammar is parsed with Top Down and Bottom-Up Parser. It is claimed to be more suitable to parse grammatical productions. it had been analyzed by Dhanashree Kulkarni et al. in 2014 [34].

This paper explored Malt and MST parsers and developed best models, which they considered because the baseline models for his or her approach. They showed that an easy system like combining both MST and Malt in an intuitive way can perform better than both the parsers. Their system's secured labelled attachment score of 90.66% and 80.77% for gold standard and automatic tracks respectively. It was presented by B. Venkata S. kumari et al in 2012 [100].

An effort towards incorporating external knowledge from Hindi Word Net to assist dependency parsing. The work is driven by the insight that idea ontologies capture a selected world aspect of lexical items, which is sort of distinct and unlikely to be deduced from morph syntactic information like morph, POS-tag and chunk. They achieve an improvement of 1.1% (LAS) when training 1,000 sentences and 0.2% (LAS) on 13,371 sentences over the baseline. The idea was given by Sambhav jain et al. in 2013 [79].

An algorithm for local word grouping to extricate fixed ordering dependencies for the Hindi sentences. Hindi being a free order language and fixed order word group extraction is important for decreasing the load on the free ordering parser. A part of speech tagging is an important

requirement for local word grouping. They present another algorithm for a part of speech tagging supported lexical sequence constraints in Hindi. It was analysed by Pradipta Ranjan et al. in 2003[61].

The different parameters of knowledge driven Malt Parser beside the two-stage pre-processing approach to create a top quality dependency parser for Hindi. The system achieved best LAS of 90.99% for gold standard track and runner-up LAS of 83.91% for automated data. This analysis was explored by Karan Singla et al. [88].

A statistical parsing of Tamil sentences using sentence structure hybrid language model. That they had built a statistical language model supported Trigram for Tamil language with medium of 5000 words. Within the experiment they showed that statistical parsing gives better performance through trigram probabilities and enormous vocabulary size. Two test cases with 120 and 40 sentences are selected from trained set and test set respectively. They reported that, the performance of the system is best than the grammar model. It was explored by Selvam M et al. in 2009 [85].

This paper is an effort at exploring and isolating some crucial cues present within the language which lend themselves to robust dependency parsing. Within the process of those experiments they also compared the performance of two freely available dependency parsers and pointed their strengths and weaknesses. in parsing. Especially note that conjoined vibhakti-label feature and minimal semantics can cause drastic improvement within the parser performance. It was presented by Bharati, Akshar, et al. in 2008 [19].

Graph-based Arabic NLP Techniques proposed for Arabic language that use graph theories and algorithms as a main tool to realize application's objective. A Graph, used to represent many sorts of knowledge like networks, webpages, social relations and text components, may be a collection of vertices (also called nodes) and connecting edges. It's classified consistent with edges properties into many various categories, such as: weighted or unweighted graphs, directed or undirected graphs, and cyclic or acyclic graphs. They presented the state-of-the-art in graph-based methods

want to handle Arabic NLP problems and applications. Distinguishing between sorts of graph used and core techniques. They introduce the key details of every method and explore how graph might be wont to gain a far better leads to Arabic NLP field. Many further works might be drained using graph in Arabic NLP applications such as: using hybrid approaches, which use graph and other techniques (like: statistical, neural networks, mathematical logic, etc.) so as to induce better results and overcome single technique limitations. Future and possible new directions might be utilized in term of using graph and deep learning techniques so as to boost Arabic NLP application's performance. It had been given by Wael Etaiwi and Arafat Awajan in 2018 [102].

Four categories of approaches to semantic analysis in NLP are (1) Distributional (2) Frame-Based (3) Model-Theoretical Approach (4) Interactive Learning. They say that the text mining to form the interaction between human and computer, though its purpose is to possess interaction among natural language of citizenry and computers. It had been represented by Monika Kanojiya and Samta Tembhekar in 2017 [96].

The research topics on natural language processing sometimes overlap with some AI and Deep Learning topics. These approaches generally adopted recently to perform NLP tasks in most effective way. The ACL 2018 Main Conference invited papers in 21areas which are Dialogue and Interactive Systems, Discourse and Pragmatics, Document Analysis, Generation, Information Extraction and Text Mining, Linguistic Theories, Cognitive Modelling and Psycholinguistics, Machine Learning, MT , Multidisciplinary, Multilinguality, Phonology, Morphology and Word Segmentation, Question Answering, Resources and Evaluation, Sentence-level Semantics, Sentiment Analysis and Argument Mining, Social Media, Summarization, Tagging, Chunking, Syntax and Parsing, Textual Inference and Other Areas of Semantics, Vision, Robotics, Multimodal, Grounding and Speech. Also represents that the economic application on NLP are often broadly classified into 3 categories: Conversational systems, Text Analytics, MT. It had been justify by Krishna Prakash Kalyanathaya, D. Akila and P. Rajesh in 2019 [67].

The impact of computer use of Natural Languages will have a profound an impression on society as would the breakthroughs in superconductors, inexpensive fusion or biotechnology. The impact of NLP by machine are going to be greater than the impact of microprocessor technology within the last 20 years, because natural language is prime to most business, military & social activities. Therefore, the appliance of NLP has without stopping. This concept was given by Aiyesha Sadiya and Archana R Hegde [2].

The target of linguistic communication processing (NLP) is to assemble computational models of natural language for its investigation and generation. First, there's mechanical inspiration of building intelligent computer frameworks like MT systems, natural language interfaces to databases, man-machine interfaces to PCs generally, speech understanding systems, text investigation and understanding frameworks etc. Second, there's a cognitive and etymology inspiration to realize a far better insight into how individuals communicate using tongue (NL). The target of the natural language Processing (NLP) is to accomplish human-like language processing, plan and assemble programming which will analyses, understand, and make languages that individuals use naturally, in order that eventually you'll be ready to address your PC as if you were addressing another person. The present voice frameworks still need adjustments -- some can’t understand heavy accents, speech impediments or quiet voices. If the info frameworks group reacts to the challenge by building NLP frameworks with reusable parts via open source programming, the longer term of NLP will start looking indeed brighter. There are still unresolved challenges for programming programs to represent everything knowledge, the differing connections and cultures of the planet. This was represented by N. Vasunthira Devi and Dr. R. Ponnusamy in 2018 [98].

CHAPTER 3

Part of Speech Tagging Using Hidden Markov Model

3.1 Introduction:

Part-of-speech tagging (or just tagging for short) is that the method of assigning a part-of speech to each word during a corpus. Tags are also usually applied to punctuation markers thus tagging for linguistic communication is that a similar processes tokenization for computer languages. Tags for natural languages are much more ambiguous. Taggers play an increasingly important role in speech recognition, natural language parsing and information retrieval as discussed in Chapter 1.

The input to a tagging algorithm may be a string of words and a specified tagset of the type described within the previous section. The output may be a single best tag for every word. For instance, here are some sample sentences from the ATIS corpus of dialogues about air-travel reservations. For each, a possible tagged output using the Penn Tree bank tagset is shown.

Book/VB that /DT flight/NN ./PUNCT

Does/VBZ that/DT/flight/NN serve/VB dinner/NN?/PUNCT

Even in these simple examples, automatically assigning a tag to every word isn't trivial. For instance, book is ambiguous. That is, it's quite one possible usage and a part of speech. It is often a verb (as in book that flight or to book the suspect) or a noun (as in hand me that book, or a book of matches). Similarly, which will be a determiner (as in Does that flight serve dinner), or a complementizer (as in i assumed that your flight was earlier). The matter of POS-tagging is to resolve these ambiguities, choosing the right tag for the context. Part-of-speech tagging is thus one among the various disambiguation tasks.

3.2 Hidden Markov Model POS Tagging:

3.2.1 Markov Chains:

The Hidden Markov Model is one among the most important machine learning models in speech and language processing [6]. To define it properly, we had wish to first introduce the Markov chain, sometimes called the observed Markov model. Markov chains and Hidden Markov Models are both extensions of the finite automata. A weighted finite-state automaton may be a simple augmentation of the finite automaton within which each arc is related to a probability show how likely that path is to be taken. The probability on all the arcs leaving a node must sum to 1.A Markov chain may be a special case of a weighted automaton during which the input sequence uniquely determines which states the automaton will undergo. Because they can’t represent inherently ambiguous problems, a Markov chain is simply useful for assigning probabilities to unambiguous sequences. Fig 3.1 shows a Markov chain for assigning a probability to a sequence of activities perform by Alexa, when she is unhappy, she is doing three activities like SLEEPS, RUNS, and EAT ICE-CREAM. This Markov chain should be familiar; actually, it represents a bigram language model, because the name suggests, the bigram model approximates the probability of a word given all the previous words by using only the conditional probability of 1 preceding word. And so, after we use a bigram model to predict the conditional probability of next word, we are thus making the following approximation:

$$p(w_n/w_1^{n-1}) \approx p(w_n/w_{n-1})$$

This assumption that the probability of a word depends only on the previous word is also known as Markov assumption. Markov models are the class of probabilistic models that assume that we can predict the probability of some future unit without looking too far in the past. Given the model in Fig.3.1, probabilities can be assigned from the vocabulary. The process of doing this can be seen in further sections.

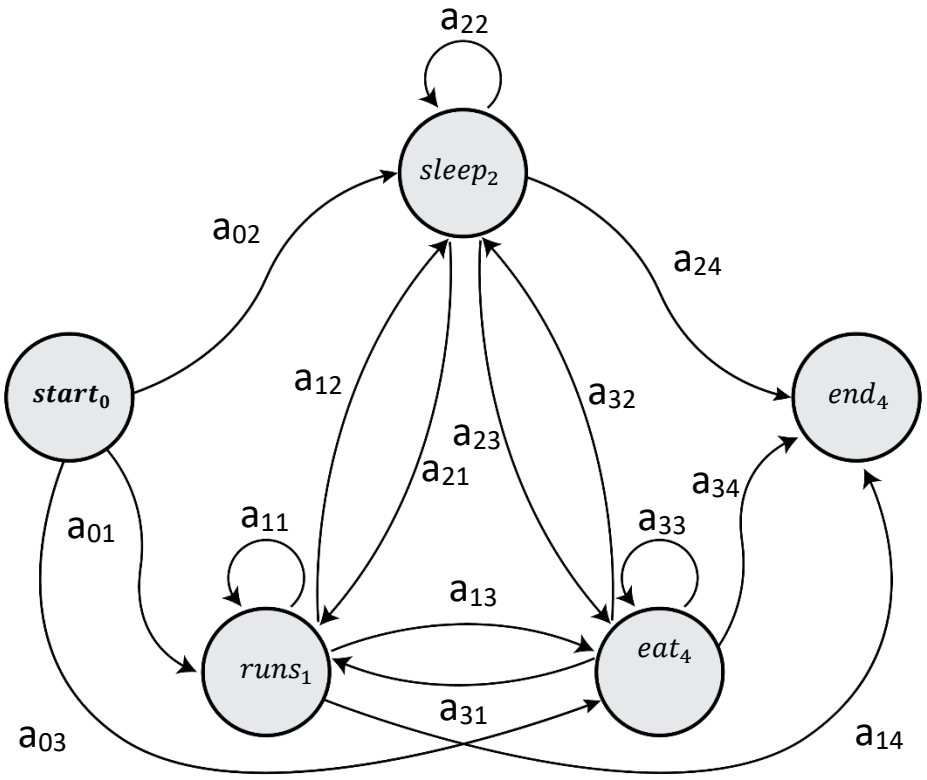


FIGURE 3.1: Simple Markov chain for the Alexa's Mood

Fig. 3.1 shows a Markov chain for assigning a probability to a sequence of activity events, for which vocabulary consist of SLEEPS, RUNS and EAT ICE-CREAM.

Markov chain can be viewed as a probabilistic graphical model; a way of representing probabilistic assumptions in a graph. A Markov chain is specified by the following components:

$Q = q_1 q_2 \dots q_N$	a set of N states
$A = a_{01} a_{02} \dots a_{n1} \dots a_{nn}$	a transition probability matrix
$q_{0,} q_F$	a special start state and final state that are not associated with Observation.

Fig. 3.1 shows represent the states (including start and end states) as nodes in the graph, and the transitions as edges between nodes.

Markov Assumption: $P(q_i / q_1 \dots q_{i-1}) = P(q_i / q_{i-1})$ (3.1)

Because each a_{ij}expresses the probability $p(q_j / q_i)$ the laws of probability require that the values of the outgoing arcs from a given state must sum to 1:

$\sum_{j=1}^{n} a_{ij} = 1 \quad \forall\, i$ (3.2)

An alternative representation that is sometimes used for Markov chains doesn't rely on a start or end state, instead representing the distribution over initial states and accepting states explicitly:

$\emptyset = \emptyset_1 \emptyset_2 \dots \emptyset_N$	An initial probability distribution over states.$\emptyset_i$is the Probability that the markov chain will start in state i. Some States j may have $\emptyset_j = 0$,meaning that they cannot be initial States. Also $\sum_{i=1}^{n} \emptyset_i = 1$
$QA = \{q_x, q_y \dots .\}$	a set $QA \sqsubset Q$of legal accepting states.

Thus, the probability of state 1 being the first state can be represented either as a_{01}or as $\emptyset_1$. Because each $\emptyset_i$expresses the probability $p(q_i/START)$, all the $\emptyset$ probabilities must sum to 1:

$$\sum_{i=1}^{n} \emptyset_i = 1 \quad (3.3)$$

Before going further, use the sample probabilities in Fig. 3.2(b) to compute the probability of each of the following sequences:

run run run run (3.4)

sleeps runs sleeps runs (3.5)

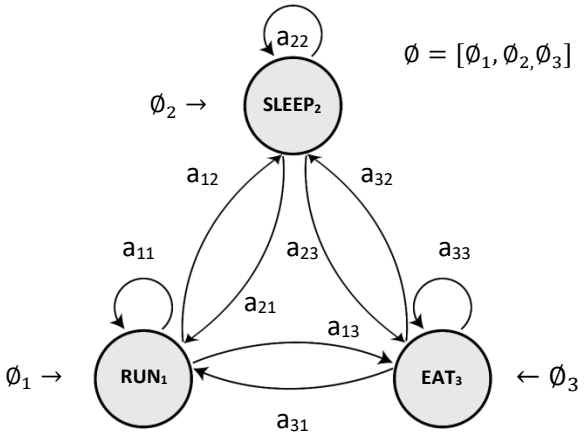


FIGURE 3.2(a)

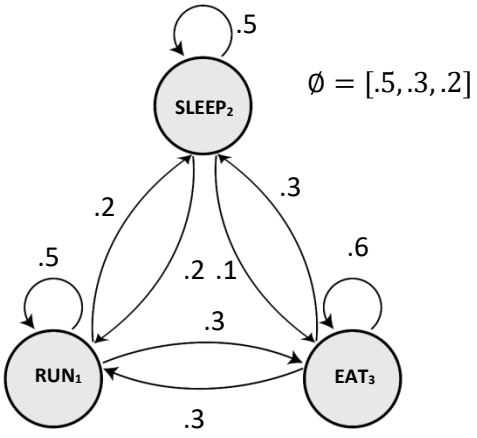


FIGURE 3.2 (b)

FIGURE 3.2: Another representation of the same Markov chain for activity shown in Fig. 3.1 Instead of using a special start state witha_{01} transition probabilities; $\emptyset$ vector is used, which represents the distribution over starting probabilities. The figure in (b) shows sample probabilities.

3.2.2 Hidden Markov Model:

A Markov chain is beneficial when there's a requirement to compute a probability for a sequence of events which will be observed in the world. In many cases, the events we have an interest in

might not be directly observable within the world. For instance, for part-of speech tagging, a part of speech tags can't be observed within the world; words are often seen, and had to infer the right tags from the word sequence. So, a part of speech tags is hidden because they're not observed. An equivalent thing are often seen in speech recognition; acoustic events are often seen within the world, and need to infer the essence of 'hidden' words that are the underlying causal source of the acoustics. A Hidden Markov Model (HMM) allows to talk about both observed events (like words which will be seen within the input) and hidden events (like part-of-speech tags). Here the observation may be a sequence of word(sentences) and it's our job to assign them a sequence of a part of speech tags [10]. Use of HMM to try to a part of speech tagging as, we define it, may be a special case of Bayesian inference, a paradigm that has been known since the work of Bayes. Selecting the tag sequence of length n that's the foremost probable given the input word sequence:

• Probabilistic generative model for sequences.
• Assume an underlying set of hidden states within which the models are often (e.g. parts of speech).
• Assume probabilistic transitions between states over time
• Assume a probabilistic generation of tokens from states
• Assume the present event depends only informed the previous event (Bigram Model). We have an input sentence

$x = x_1, x_2, x_3 \dots \dots . . x_n$ (x_i is the i^{th} word in the sentences)

We have a tag sequence

$y = y_1, y_2 y_3 \dots \dots \dots . y_n$ (y_i is the i^{th}tag in the sentences)

In other words, we want , out of all sequences of n tags y the single tag sequences such that $p(y/x)$is highest.

We will use HMM defined as

$$p(x_1, x_2, x_3 \dots \dots . . x_n, y_1, y_2 y_3 \dots \dots \dots . y_n)$$

for any sentences $x_1, x_2, x_3 \dots \dots . . x_n$ and tag sentences $y_1, y_2 y_3 \dots \dots \dots . y_n$ of the same length.

Then the most likely tag sequence for x is ,

$$arg \underset{y_1,y_2,y_3.....y_n}{max} p(x_1, x_2, x_3 \dots \dots \dots x_n, y_1, y_2 y_3 \dots \dots \dots \dots y_n) \quad (3.6)$$

[The function $\underset{x}{argmax} f(x)$ means "the x such that f(x) is maximized."]

The intuition of Bayesian classification is to used Bayes' rule to transform Eq. (3.6) into a set of other probabilities, which turns out to be easier to compute. Bayes 'rule is presented by Eq. (3.7) it gives us how to interrupt down any conditional probability $p(y/x)$into three other probabilities:

$$p(y/x) = \frac{p(x/y)p(y)}{p(x)}$$

$$\frac{p(x_1,x_2,x_3.....x_n,y_1,y_2y_3.....y_n)p(y_1,y_2y_3.....y_n)}{p(x_1,x_2,x_3.....x_n)} \quad (3.7)$$

$here\ x_1, x_2, x_3 \dots x_n = p(s)$, which is fixed so it is negligible.

Now,

$$p(x_1, x_2, x_3 \dots \dots \dots x_n, y_1, y_2 y_3 \dots \dots \dots \dots y_n) p(y_1, y_2 y_3 \dots \dots \dots \dots \dots \dots \dots \dots \ y_n) \quad (3.8)$$
$$p(x_1, x_2, x_3 \dots \dots \dots x_n, y_1, y_2 y_3 \dots \dots \dots \dots y_n)$$

$$= p(x_1/y_1, y_2 y_3 \dots \dots \dots \dots y_n). p(x_2/x_1, y_1, y_2 y_3 \dots \dots \dots \dots y_n). p(x_3/x_2, x_1, y_1, y_2 y_3 \dots \dots \dots \dots y_n). \dots$$

$$p(x_n/x_{n-1}, x_{n-2,.....} y_1, y_2 y_3 \dots \dots \dots \dots y_n).$$

i. $e p(x_1/y_1) p(x_2/y_2) p(x_3/y_3) \dots \dots \dots p(x_n/y_n)$

$$= \prod_{i=1}^{n} p(x_i/y_i) \quad (3.9)$$

Now

$$p(y_1, y_2 y_3 \dots \dots \dots \dots y_n)$$
$$= p(y_1). p(y_2/y_1) p(y_3/y_1\, y_2) \dots \dots \dots p(y_n/y_{n-1}\, y_{n-2} y_{n-3} \dots \dots y_1)$$
$$= p(y_1). p(y_2/y_1) p(y_3/y_2) \dots \dots \dots \dots \dots \dots p(y_n/y_{n-1}) \dots \{\text{Bigram model}\}$$
$$= \prod_{i=1}^{n} p(y_i/y_{i-1})\, p(y_1) \quad (3.10)$$

Therefore eq[n]. (3.7) becomes,

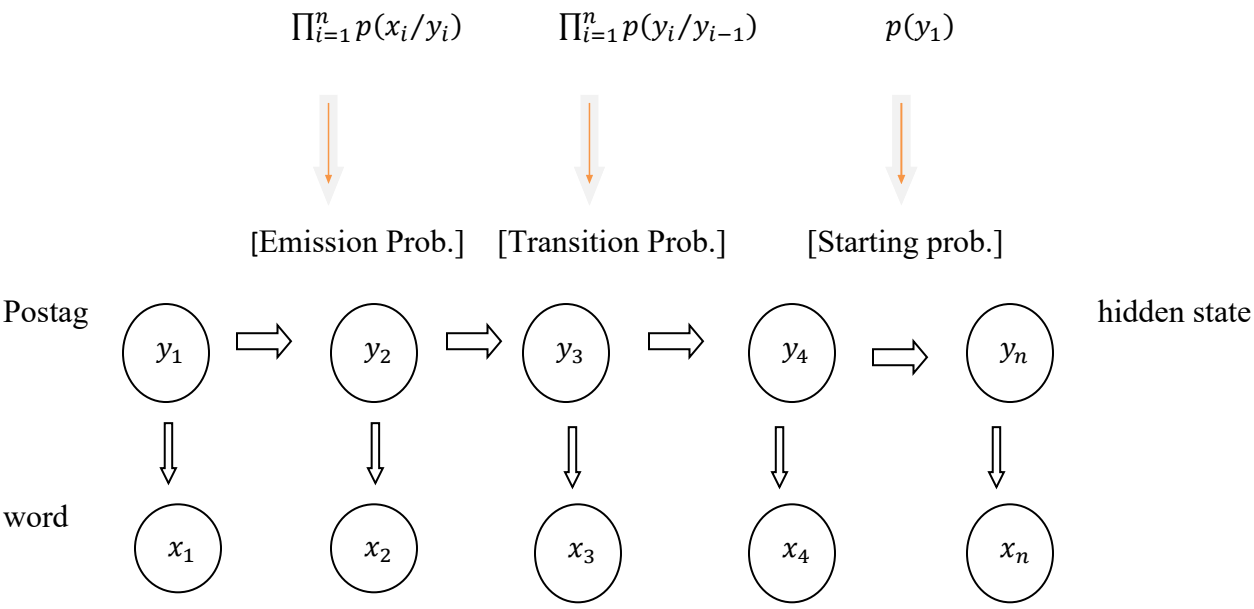


FIGURE 3.3: Formalizing HMM Taggers

A Hidden Marlov Model (HMM) allows us to talk both about Observed event (like word that we see in the input) and about Hidden events (like Part-of-speech tags) that we think of as ausal factors in our probabilistic model. The following is a formal definition of a Hidden Markov Model, explaining how it differs from a Markov chain. An HMM is specified by the following components:

$Q = q_1 q_2 \ldots q_N$	a set of N sets
$A = a_{11} a_{12} \ldots a_{n1} \ldots a_{nn}$	a transition probability matrix
$O = o_1 o_2 \ldots o_T$	a Sequence of T observation
$B = b_i(o_t)$	Emission probabilities
q_0, q_F	Start state and end (final) state

3.3 Using the Viterbi Algorithm for HMM Tagging:

For any model, like an HMM, that contains hidden variable, the task of determining which sequence of variable is that the underlying source of some sequence of observation is named decoding task. The Viterbi algorithm is probably the foremost common decoding algorithm used for HMMs, whether for part-of-speech tagging or speech recognition. The term Viterbi is common in speech and Language processing, but this is often really standard application of the classic dynamic Programming algorithm. The marginally simplified version of the Viterbi algorithm that we present takes as input one HMM and a sequence of observed words $O = (o_1 o_2 \dots o_T)$ and returns the most probable state/tag sequence $Q = (q_1 q_2 \dots q_T)$, together with its probability.

Let the HMM defined by the two tables in Table 3.1 and Table 3.2. Table 3.1 express the a_{ij}probabilities, the transition probabilities between hidden state (i.e., part-of-speech tags). Table 3.2 expresses the $b_i(o_t)$ probabilities, the observation likelihoods of words given tags

Suppose there are three sentences:

- THE STUDENT PASS THE TEST .
  Det N V Det N PUNCT
- THE STUDENT WAIT FOR THE PASS .
  Det N V P Det N PUNCT
- TEACHER TEST STUDENTS .
  N V N PUNCT

TABLE 3.1: Tag transition probabilities (the array,$p(t_i/t_{i-1})$ compute from the tag set.

	Det	**N**	**V**	**P**	**Punct.**	Q_F
q_0	$\frac{2}{3}$	$\frac{1}{3}$	0	0	0	0
Det	0	1	0	0	0	0
N	0	0	$\frac{1}{2}$	0	$\frac{1}{2}$	0
V	$\frac{1}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$	0	0
P	1	0	0	0	0	0
Punct.	0	0	0	0	0	1

TABLE 3.2: Observation likelihoods (the b array) computed from the tag set (Emission Probabilities)

	THE	**STUDENT**	**PASS**	**TEST**	**WAIT**	**FOR**	**TEACHER**	**.**
Det	$\frac{4}{4}=1$	0	0	0	0	0	0	0
N	$\frac{0}{6}=0$	$\frac{3}{6}=\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{0}{6}=0$	$\frac{0}{6}=0$	$\frac{1}{6}$	$\frac{0}{6}=0$
V	$\frac{0}{3}=0$	$\frac{0}{3}=0$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{0}{3}=0$	$\frac{0}{3}=0$	$\frac{0}{3}=0$
P	$\frac{0}{1}=0$	0	0	0	0	1	0	0
Punct.	$\frac{0}{3}=0$	0	0	0	0	0	0	$\frac{3}{3}=1$

Fig. 3.4 Shows pseudocode for the Viterbi algorithm. The Viterbi algorithm sets up a probability matrix, with one column for every observation t and one row for every state within the state graph.

Function VITERBI (observation of len T, state-graph of len N) returns best-path

create a path probability matrix Viterbi [N+2,T]

for each state s from 1 to N do ; initialization step

viterbi [s,1]← $a_{0,s} * b_s(o_1)$

back Pointer [s,1]← 0

for each time step t from 2 to T do ; recursion step

for each state s from 1 to N do

viterbi [s,1]← $max_{s'=1}^{N}$ viterbi $[s', t-1] * a_{s',s} * b_s(o_t)$

backpointer [s,t]← $argmax_{s'=1}^{N}$ viterbi $[s', t-1] * a_{s',s}$

viterbi $[q_{F,} T]$]← $max_{s=1}^{N}$ viterbi [s,T]*a_{s,q_F} ; termination step

back pointer $[q_{F,}T]$ ← $argmax_{s=1}^{N}$ viterbi [s,T]*a_{s,q_F} ; termination step

return the back trace path by following back pointers to states back in time from back pointer $[q_{F,}T]$

FIGURE 3.4: Viterbi algorithm for finding optimal sequence of tags.

3.4 Experimental Procedure for Gujarati text:

Experiment is administered using Matlab. The particulars of the inputs to the Matlab code is given below:

• Total tags : 28

• Database: 351 Guajarati words

• Size of transition matrix: 28x 28

• Size of Emission matrix: 28 x 351

The transition and emission matrices are computed by implementing in Matlab, a programming language. The Tables 3.3 and 3.4 represent the classification accuracy and error analysis whether the mismatch takes place respectively. The Tag wise error graph is shown in Fig. 3.5.

3.4.1 Classification accuracy:

TABLE 3.3: Viterbi Accuracy for Gujarati

Techniques	No. of mismatch	Accuracy
Viterbi	25	92.87%

3.4.2 Tag Wise Error Analysis for Viterbi:

In order to improve any model, we need to understand where it went wrong. Analyzing the error in a classifier like Part of speech tagger is done with a confusion matrix. Error Analysis like this is a crucial part of any computational linguistic application. Error analysis can help find bugs, find problems in the training data, and most important, help in developing new kinds of knowledge or algorithms to use in solving problems.

TABLE 3.4: Error Analysis Table

ORIGINAL WORD NO.	ORIGINAL TAG	VITERBI	VITERBI	POSITION	EXACT PLACE	WORD
7	DMD	1	NN	5	10	આ
5	VM	11	VAUX	8	16	છે
8	CCS	17	CCD	30	60	કે
11	VAUX	5	VM	73	146	લાગી
11	VAUX	5	VM	111	222	રહ્યા
5	VM	11	VAUX	145	290	છે
5	VM	11	VAUX	154	308	છે
17	CCD	13	RPD	156	312	પણ
5	VM	11	VAUX	162	324	છે
7	DMD	9	DMR	164	328	આ
8	CCS	17	CCD	178	356	કે
7	DMD	9	DMR	180	360	આ
7	DMD	1	NN	194	388	એ
8	CCS	13	RPD	198	396	પણ
9	DMR	1	NN	200	400	એ
13	RPD	20	PSP	209	418	એટલે
1	NN	2	NNP	210	420	મહાજન
16	VNG	1	NN	211	422	કામે
11	VAUX	5	VM	213	426	ગયા
7	DMD	1	NN	234	468	એ
1	NN	5	VM	235	470	જગ્યા
13	RPD	20	PSP	246	492	એટલે
7	DMD	9	DMR	251	502	આ
11	VAUX	5	VM	274	548	ગયા
3	VNF	16	VNG	336	672	આવીને

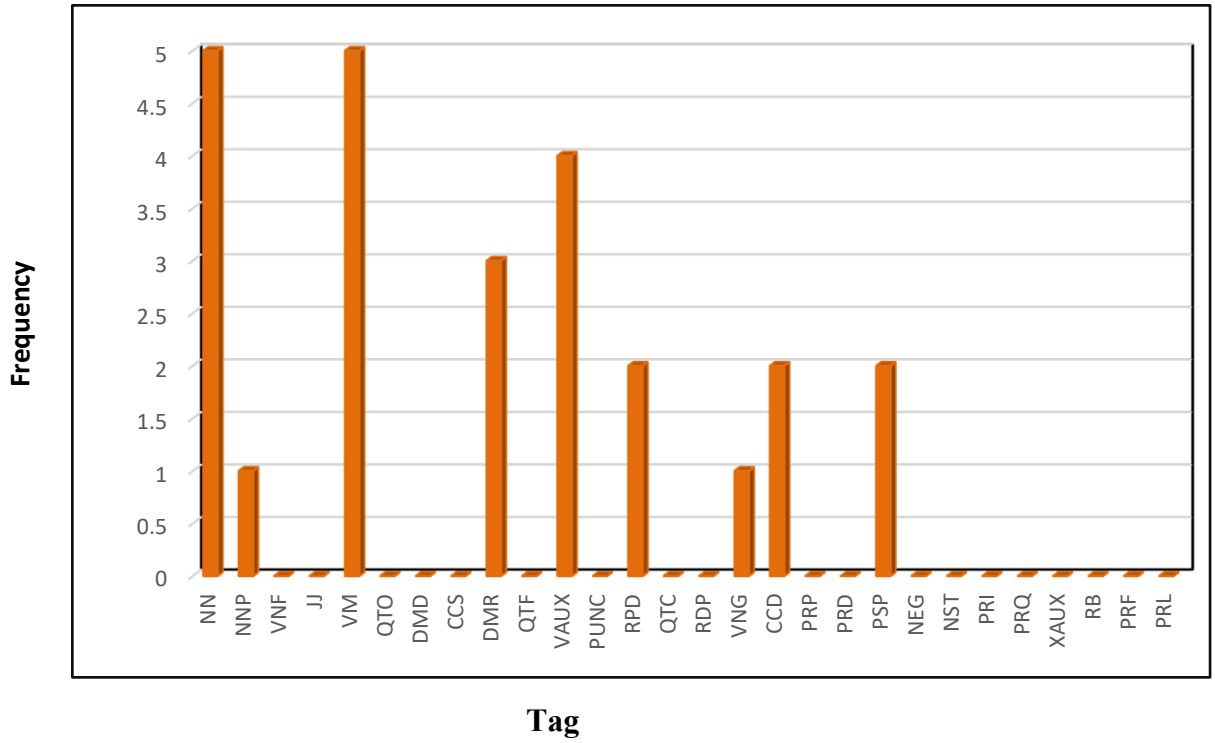


FIGURE 3.5: Graphical Representation of the Error Analysis

3.5 Conclusion:

Table 3.5 demonstrates the error analysis of the words whether mismatch takes place. In 14 cases out of 25 mismatches which is highlighted are confusing with in the same category. For ex. within the second row the first Tag is VM while the expected tag using Viterbi algorithm is VAUX for the word "ଛ". Ignoring such cases only 9 places are there which the particular mismatch takes place the accuracy is achieved around 92.87 %. The accuracy is often ameliorated by taking significantly large database.

CHAPTER – 4

Part of Speech tagging Using Support Vector Machine

4.1 Introduction:

There are different machine learning approaches to the problem of assigning each word of a text with a parts of speech tag. During this the performance of a POS Tagger for Gujarati language is shown using SVM and wavelet based SVM [62].

Support Vector Machine is essentially used for classification and recognizes the pattern [37]. SVMs have high generalization performance independent of dimension of feature vectors. Other algorithms require careful feature selection, which is typically optimized heuristically, to avoid over fitting. SVMs can perform their learning with all combinations of given features without increasing computational complexity by introducing the Kernel function. Conventional algorithms cannot handle these combinations efficiently.

Wavelets are often compared to a wide-angle optical lens that permits one to require broad landscape portraits also as zoom in on microscopic detail that is normally hidden to the human eye. In mathematical terms, wavelets are local orthonormal bases consisting of small waves that dissect a function into layers of different scale. Wavelet theory has its roots in Fourier analysis, but there are important differences. The Fourier transformation uses a sum of sine and cosine functions at different wavelengths to represent a given function. Sine and cosine functions,

however, are periodic functions that are inherently nonlocal, they are going on to plus and minus infinity on both ends of the real line. Therefore, any change at a specific point of the time domain has an impact that's felt over the entire real line. In praxis, this suggests that we assume the frequency content of the function to be stationary along the time axis. To overcome this restriction, researchers invented the windowed Fourier transform. The data are cutup into several intervals along the time axis and therefore the Fourier transform is taken for every interval separately. Wavelets, on the opposite hand, are defined over a finite domain. More importantly, wavelets can cut data up into different frequency components for individual analysis. This scale decomposition opens an entire new way of processing data. Wavelet basis consists of a father wavelet that represents the graceful baseline trend and a mother wavelet that's dilated and shifted to construct different levels of detail. This resembles the building plan of a natural organism that's supported self-similarity. At high scales, the wavelet features a small-time support, enabling it to zoom in on details like spikes and cusps, and on short-lived phenomena like delta functions. At low scales, wavelets capture long-run phenomena. Their ability to adapt their scale and time support enables them to escape Heisenberg's curse; i.e., the law that says that one can't be simultaneously precise within the time and therefore the frequency domain.

4.2 Support Vector Machine:

The support vector machine (SVM) may be a universal constructive learning procedure based on the statistical learning theory. A support vector machine (SVM) may be a supervised learning technique from the field of machine learning applicable to both classification and regression. Originally it had been worked out for linear two-class classification with margin, where margin means the minimal distance from the separating hyperplane to the closest data points. SVM learning machine seeks for an optimal separating hyperplane, where the margin is maximal. A crucial and unique feature of this approach is that the solution based only on those data points, which are at the margin. These points are called support vectors [25,43]. There are 2 sorts of SVM classifiers:

1. Linear SVM Classified
2. Non-Linear SVM Classifier

The role of SVM in NLP is applied to text categorization, and provides the high accuracy with a large number of texts taken as features. We defining very simple case, a two-class problem where the classes are linearly separable [86].

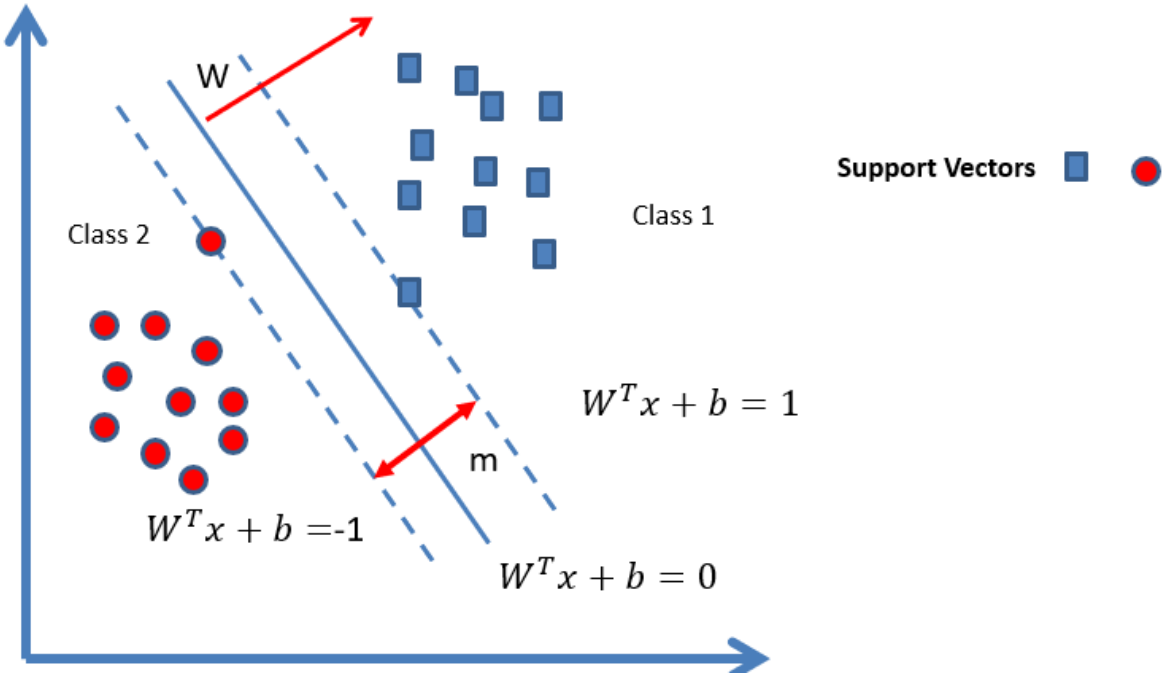


FIGURE 4.1 Illustration of the idea of an optimal hyper plane

4.2.1 Optimum Hyperplane for Linearly Separable:

Given training a sample, the support vector machine construct a hyper plane as the decision surface in such a way that the margin of separation between positive and negative example is maximized. Let the data set D be given as $\{x_i, d_i\}_{i=1}^{N}$where $x_i$is the set of input pattern with associated class labels d_i. Each d_i cantake one of two values, either +1 or -1(i.e., $d_i\{+1,-1\}$).An SVM approach this problem by searching for the Maximum Marginal Hyperplane. SVM searches for the hyperplane with the largest margin, that is, the Maximum Marginal Hyperplane. The associated margingives the largest separation between classes. A separating hyperplane can be written as $w^T x + b = 0$, Where $w$is a weight vector and b is a bias.

Hyperplane:$g(x) = w^T x + b$,w is a adjustable weight vector and b is a bias.

Lemma 4.1 Distance from x to the plane r=$\frac{g(x)}{\|w\|}$.

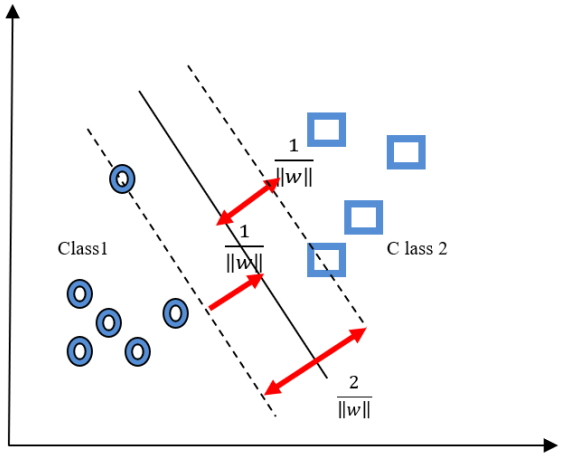


FIGURE 4.2 Maximum margin linear classifier

$\boldsymbol{g(x) = w^T x + w_0 = 0}$
$\boldsymbol{x = x_p + r\frac{w}{\|w\|}}$

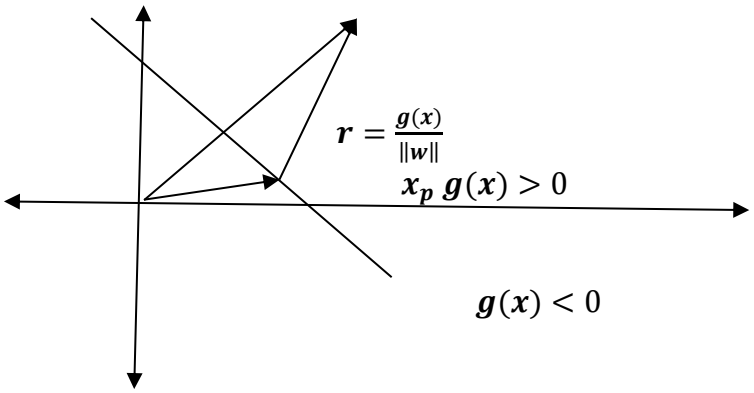


FIGURE 4.3 Geometric interpretation of algebraic distance of points to the optimal hyper plane for a two-dimensional case

Proof : Let $x = x_p + r\frac{w}{\|w\|}$ where r is the distance from x to the plane=0.

Here $g(x) = w^T x + b \Longrightarrow g(x) = w^T(x_p + r\frac{w}{\|w\|}) + b$

$$= w^T x_p + w^T r \frac{w}{\|w\|} + b$$

$$= w^T x_p + b + w^T r \frac{w}{\|w\|}$$

$$= g(x_p) + r \frac{\|w\|^2}{\|w\|} \quad [w.w^T = \|w\|^2]$$

$$= 0 + r.\|w\|$$

$$= r.\|w\|$$

Thus the maximum margin $m = \frac{2}{\|w\|}$.

Hence the proof.

4.2.1.1 Quadratic Hyperplane:

SVM concepts for typical two-class classification problems can be discussed for explanation. Given a training set of instancelabel pairs

$$(x_i , y_i) \ , i = 1,2, \dots \dots m, \ x_i \in R^n \text{ and } y_i \in \{+1 \ , -1\}$$

The geometric representation algebraic distance of points the pair (x_i, y_i) must satisfied the following constraint:

$w_0^T x_i + b_0 \geq 1 \quad for\ y_i = +1 \quad \dots\dots\dots$ (4.1)

$w_0^T x_i + b_0 < 1 \quad for\ y_i = -1 \quad \dots\dots\dots\dots$ (4.2)

Combining Eqs. (1) and Eqs. (2) into one set of inequalities.

$$y_i(w^T x + b) \geq 1 \quad \forall \ i = 1,2 \dots . . m \tag{4.3}$$

The SVM finds an optimal separating hyperplane with the maximum margin by solving the following optimization problem:

subject to: $y_i(w^T x + b) \geq 1 \quad \forall\ i = 1,2 \ldots . . m$ (4.4)

and the weight vector w minimize the cost function

$$\emptyset(w) = \frac{1}{2} w^T . w$$

the scaling function $\frac{1}{2}$ included here for convenience of presentation. It is called Primal problem. It is known that to solve this quadratic optimization problem one must find the saddle point of the Lagrange function:

$L_p(w, b, \alpha) = \frac{1}{2} w^T . w - \sum_{i=1}^{m} \alpha_i \ [y_i(w^T x + b) - 1] \qquad \alpha_i \geq 0$ (4.5)

{Primal optimization}

Where, α_i the denotes Lagrange multipliers, hence $\alpha_i \geq 0$ The search for an optimal saddle point is necessary because the Lp must be minimized with respect to the primal variables w and b are maximized with respect to the nonnegative dual variable α_i .

By differentiating with respect to w and b, and setting the result equal to zero, the following equations are obtained

$$\frac{\partial}{\partial w} L_p = 0 \ , \ w = \sum_{i=1}^{m} \alpha_i y_i x_i \quad (4.6)$$

$$\frac{\partial}{\partial b} L_p = 0 \ , \ \sum_{i=1}^{m} \alpha_i y_i = 0 \quad (4.7)$$

The Karush Kuhn–Tucker (KTT) conditions for the optimum constrained function are necessary and sufficient for a maximum of Eq. (5). The corresponding KKT complementarity conditions are

$$\alpha_i [y_i(w^T x + b) - 1] \ = 0, \ \forall i \quad (4.8)$$

The primal problem deals with a convex cost function and linear constraint. Given such Constraint -optimization problem ,it is possible to construct the another problem called dual problem.

4.2.2 Optimal Hyperplane for Non-Separable:

The solution for the non-separable case is to introduce slack variables ξ_i that relax the constraints of the canonical hyperplane equation

$y_i(w^T x + b) \geq 1 - \xi_i$ for i=1,2,…….N $\xi_i \geq 0$ for all i …… (4.9)

ξ_i are called slack variable; they measure deviation of a data point from the ideal condition of pattern separability.

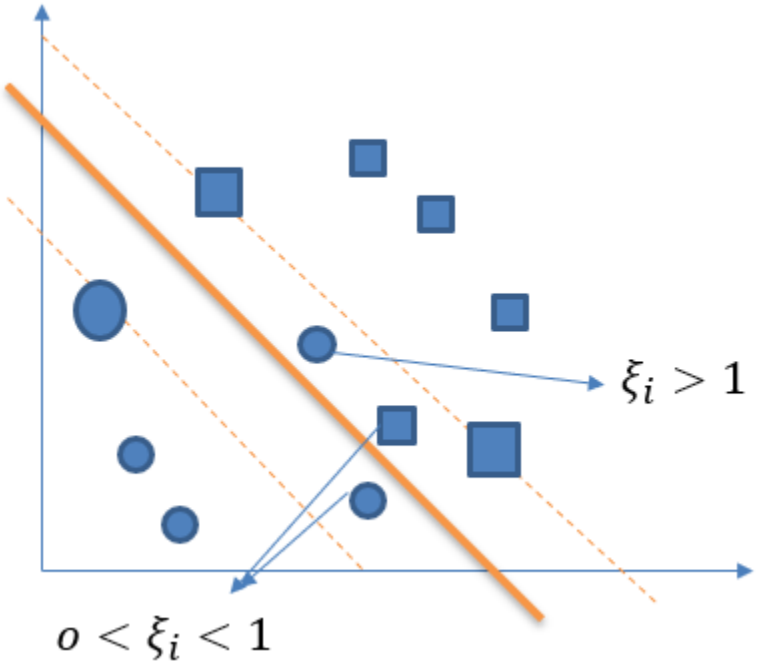


FIGURE 4.4 SVM for non-linearly separable case

$\xi_i > 1$ sample i is on the wrong side of the separating hyperplane.$o < \xi_i < 1$ sample i is on the right side of separating hyperplane but within the region of maximum margin. $\xi_i < 0,$ is the ideal case for sample i.We may do this minimization function .

$\emptyset(\xi) = \sum_{i=1}^{N} I(\xi_i - 1)$ with respect to the weight vector w, subject to the constraint in eqn (1) and constraint on $\|w\|^2$.

The $I(\xi)$ is an indicator function defined by $I(\xi) = \begin{cases} 0, & \xi \leq 0 \\ 1, & \xi > 0 \end{cases}$

Minimization of $\emptyset(\xi)$ with respect to w is a non convex optimization problem that is NP complete.

i.e, $\emptyset(\xi) = \sum_{i=1}^{N} \xi_i$

therefore the functional to be minimization w.r.t. the weight vector

$$\emptyset(w, \xi) = \frac{1}{2} w^T . w + C \sum_{i=1}^{N} \xi_i \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.10)$$

The Parameter C controls the tradeoff between complexity of the machine and the number of non separable points.

Note that neither the slack variable ξ_i nor their own Lagrange multiplier appear in the dual problem. In this formulation of the optimization problem the training data only occur in inner products of the feature vectors. This allows the use of kernel function.

4.3 Kernel Function:

In some cases, no separating hyper plane can be found in the given vector space. The application of a feature map $\emptyset : R^n \rightarrow R^N$ Projecting the training points into a high dimensional feature space where the data is linearly separable often leads to increased computational complexity referred to as dimensionality. SVMs can handle such high dimensional feature space because they employ

kernel function. A kernel function provides a way to compute efficiently inner product of two vectors in the features space without explicit calculation of the feature mapping [43,52,104],

$$k(x,y) =< \emptyset(x).\emptyset(y) >$$

Every kernel holds the non-linear kernel function. Polynomial kernel $K(x,y) = (x.y)^d$, d dimensions.

4.3.1 A Polynomial Kernel Function:

Suppose $x = (x_1, x_2)$ is the input vector.

$$K(x,y) = (x^T.y)^2$$

Here $(x^T.y)^2 = \varphi(x)^T.\ \varphi(y)$(we have to find this)

The feature space mapping is: $\varphi(x) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)^T$

Then inner product is

$\varphi(x)\varphi(y) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)^T\ \ (y_1^2, \sqrt{2}y_1y_2, y_2^2)$

$=(x_1y_1 + x_2y_2)^2$

Polynomial kernel function to compute the same value is

$$K(x,y) = (x^T.y)^2$$

$= ((x_1, x_2)^T.(y_1, y_2))^2$

$=(x_1y_1 + x_2y_2)^2$

$K(x,y) =\ \varphi(x)^T.\ \varphi(y)$

4.3.2 Kernel Method:

Let x denote a vector from the input space of dimension m_0 .

Let $\{\emptyset_j(x)\}_{j=1}^{\infty}$ denote a set of non linear function that between them we may define a hyperplane acting as a decision surface with the formula

$$\sum_{j=1}^{\infty} w_j \emptyset_j(x) = 0 \quad (4.11)$$

Where $\{w_j\}_{j=1}^{\infty}$ denote a infinitely large set of weights that transform the feature space to the output space. Using matrix notation equation (4.1) rewrite as

$$w^T \emptyset(x) = 0 \quad (4.12)$$

Where $\emptyset(x)$a feature vector and w is the corresponding weight vector.

Linear separability of the transform pattern in the feature space we may write ,

$w = \sum_{i=1}^{N_s} \alpha_i y_i \emptyset(x_i)$, $N_s$is the number of support vectors. (4.13)

Where the feature vector is expressed as $\emptyset(\boldsymbol{x_i}) = [\emptyset_{\boldsymbol{1}}(\boldsymbol{x_i}), \emptyset_{\boldsymbol{2}}(\boldsymbol{x_i}), \ldots.]^{\boldsymbol{T}}$ (4.14)

substitute Eq. (4.2) into Eq. (4.3), the decision surface in the output space

$$\sum_{i=1}^{N_s} \alpha_i y_i \, \emptyset^{\boldsymbol{T}}(\boldsymbol{x_i})\emptyset(\boldsymbol{x}) = 0 \quad (4.15)$$

Accordingly, $\boldsymbol{k(x, x_i)} = \emptyset^{\boldsymbol{T}}(\boldsymbol{x_i})\emptyset(\boldsymbol{x}) \quad \{Kernel\ Trick\}$

$$= \sum_{\boldsymbol{j=1}}^{\infty} \emptyset_{\boldsymbol{j}}(\boldsymbol{x_i})\emptyset_{\boldsymbol{j}}(\boldsymbol{x}) \quad , \boldsymbol{i = 1, 2 \ldots\ldots N_s}$$

We may express the optimal decision surface in output space

$$\sum_{\boldsymbol{i=1}}^{\boldsymbol{N_s}} \boldsymbol{\alpha_i y_i} \; \boldsymbol{k(x, x_i) = 0}$$

The function $\boldsymbol{k(x, x_i)}$ is called a inner product kernel or simply the kernel it has two properties:

1. The Kernel is symmetric about the centers point
2. The total volume under the surface of the function kernel is constant

$k(x_i, x_j)$ as the ij-th element of the symmetric N by N- matrix

$$K = \{ k(x_i, x_j) \}_{i,j=1}^{N} \quad (4.16)$$

The matrix K is a non negative definite matrix called Kernel Matrix, It is called as a Gram.

Theorem 4.1 Mercer's theorem: Let $k(x, x')$ be a continuous symmetric kernel that is defined in the closed interval $a \leq x \leq b$, and likewise for x'. The kernel $k(x, x')$ can be expanded in the series $k(x, x') = \sum_{i=1}^{\infty} \lambda_i \phi_i(x) \phi_i(x')$ with positive coefficient $\lambda_i > 0$, $for\ all\ i$ .For this expansion to be valid and for it to converge absolutely and uniformly, it is necessary and sufficient

that the condition $\int_a^b \int_a^b k(x,x')\,\varphi(x)\varphi(x')dxdx' \geq 0$ holds for all $\varphi(.)$ For which we have $\int_a^b \varphi^2(x)dx < \infty$,where a and b are constant of integration [55].

The features $\varphi_i\,(x)$ are called eigen functions of the expansion, and the numbers λ_i are the eigenvalues. The fact that all of the eigenvalues are positive means that the kernel is positive definite.

4.4 Design of SVM:

We use the same optimization problem as before, and replace the dot product$\emptyset^T(x_i)\emptyset(x_j)$ with the kernel $k(x_i, x_j)$. The Lagrange dual problem for the non-linear SVM is simply

$$Q(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j k(x_i, x_j)$$

$subject\ to\ the\ constraints$

$\sum_{i=1}^{m} \alpha_i y_i$=0

$0 < \alpha_i \leq C$, $i = 1 \ldots\ldots N$

where C is the user-specific positive parameter.

Mercer's Condition only tells us whether a kernel is actually an inner-product kernel, but it does not tell us how to construct the functions $\varphi_i\,(x)$ for the expansion Table 4.1 summarize the kernel for three common types of support vector machine.

TABLE 4.1: Summaries of Mercer's Kernels

Type of SVM	**Mercer kernel** $\boldsymbol{k(x, x_i), i = 1, 2 \ldots \ldots N}$	**Comments**	**Where it used**
Polynomial learning Machine	$(x^T x_i + 1)^p$	Power p is specified a priori by the user	It is popular in image processing.
Radial basis-function network (Gaussian Kernel)	$\exp\left(-\frac{1}{2\sigma^2}\|x - x_i\|^2\right)$	The width σ^2, common to all the kernels.	It is a general-purpose kernel; used when there is no prior knowledge about the data
Two-layer perceptron	$\tanh\left(\beta_0 x^T x_i + \beta_1\right)$	Mercer condition only satisfied for some β_0 and β_1	We can use it as the proxy for neural networks.

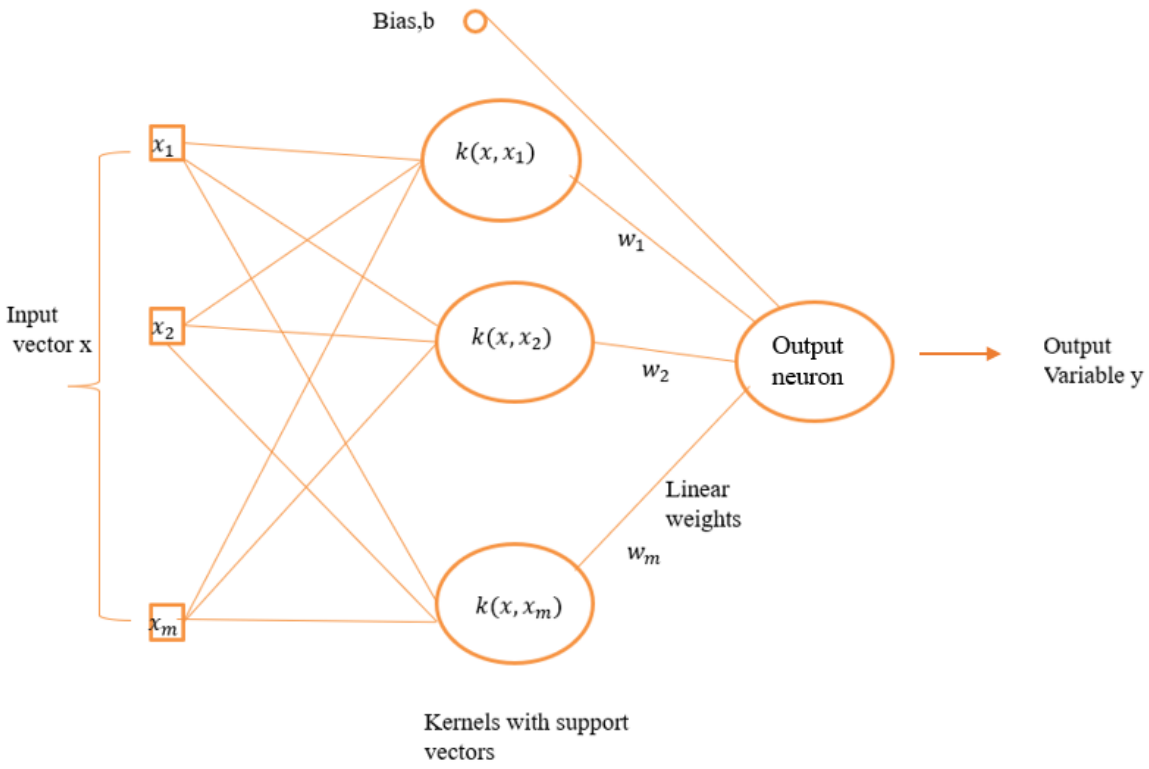


FIGURE 4.5: Architecture of support vector machine

4.5 XOR Problem:

TABLE 4.2: XOR Problem

Input vector x	Desired response d
(-1,-1)	-1
(-1,+1)	+1
(+1,-1)	+1
(+1,+1)	-1

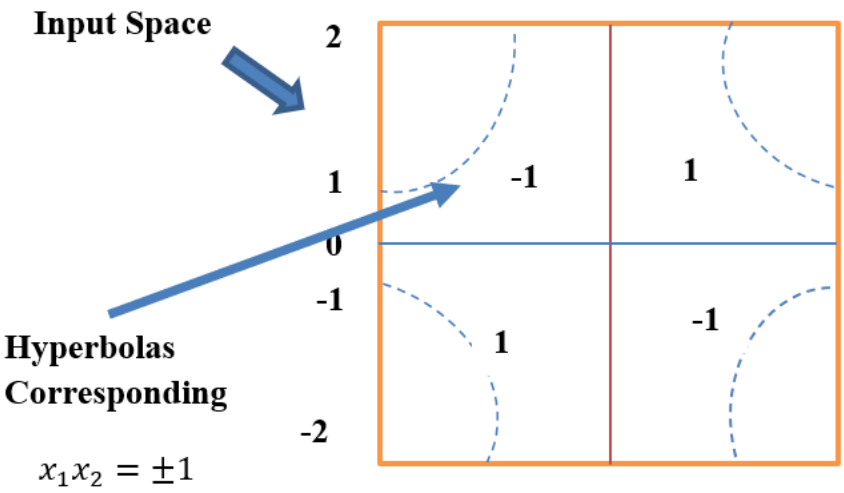


FIGURE 4.6: Decision boundary constructed by the network

- Data set:
- Class 1: $x_1 = (-1,-1) x_4 = (+1,+1)$
- Class 1: $x_2 = (-1,+1) x_3 = (+1,-1)$
- Kernel function
- Polynomial of order 2 : $k(x, x_i)$ $=(1 + x^T x_i)^2$

  Solution $x = [x_1, x_2]^T x_i = [x_{i1}, x_{i2}]$
- $k(x, x_i) = 1 + x_1^2 x_{i1}^2 + 2x_1 x_2 x_{i1} x_{i2} + x_2^2 x_{i2}^2 + 2x_1 x_{i1} + 2x_2 x_{i2}$
- Feature space: $\varphi(x) = [1, x_1^2, \sqrt{2}x_1 x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2]^T$
- $K = \{ k(x_i, x_j) \}_{i,j=1}^{N}$

- $K = \begin{bmatrix} k(x_1,x_1) & k(x_1,x_2) & k(x_1,x_3) & k(x_1,x_4) \\ k(x_2,x_1) & k(x_2,x_2) & k(x_2,x_3) & k(x_2,x_4) \\ k(x_3,x_1) & k(x_3,x_2) & k(x_3,x_3) & k(x_3,x_4) \\ k(x_4,x_1) & k(x_4,x_2) & k(x_4,x_3) & k(x_4,x_4) \end{bmatrix}$
- $k(x_1,x_1) = (1 + x_1^T x_1)^2$

  $= (1 + (-1,-1)^T(-1,-1))^2$

  $= 9$
- $K = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix}$
- Lagrange Multiplier :
- $Q(\alpha) = \sum_{\boldsymbol{i=1}}^{\boldsymbol{m}} \boldsymbol{\alpha_i} - \frac{\mathbf{1}}{\mathbf{2}} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$
- $Q(\alpha) = \sum_{\boldsymbol{i=1}}^{\boldsymbol{4}} \boldsymbol{\alpha_i} - \frac{\mathbf{1}}{\mathbf{2}} \sum_{i=1}^{4} \sum_{j=1}^{4} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$

  $=(\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4) - \frac{1}{2}[9\alpha_1^2 - 2\alpha_1\alpha_2 - 2\alpha_1\alpha_3 + 2\alpha_1\alpha_4 + 9\alpha_2^2 + 2\alpha_2\alpha_3 - 2\alpha_2\alpha_4 + 9\alpha_3^2 - 2\alpha_3\alpha_4 + 9\alpha_4^2]$
- Differentiate w.r.t $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ and put equal to zero., we get

$$9\alpha_1 - \alpha_2 - \alpha_3 + \alpha_4 = 1$$
$$-\alpha_1 + 9\alpha_2 + \alpha_3 - \alpha_4 = 1$$
$$-\alpha_1 + \alpha_2 + 9\alpha_3 - \alpha_4 = 1$$
$$\alpha_1 - \alpha_2 - \alpha_3 + 9\alpha_4 = 1$$

- It solved by Gauss Elimination Method; we get $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = \frac{1}{8}$
- **Optimum Weight**:
- $w = \sum_{i=1}^{N_s} \alpha_i y_i \emptyset(x_i)\, N_s$: number of support vectors
- $w_0 = \sum_{i=1}^{4} \alpha_i y_i \emptyset(x_i)$

  $=\alpha_1 y_1 \emptyset(x_1) + \alpha_2 y_2 \emptyset(x_2) + \alpha_3 y_3 \emptyset(x_3) + \alpha_4 y_4 \emptyset(x_4)$
- Here $\varphi(x) = [1, x_1^2, \sqrt{2}x_1x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2]^T$

$$\varphi(-1,-1) = [1,\ 1,\ \sqrt{2}, 1, -\sqrt{2}, -\sqrt{2}]^T = \begin{bmatrix} 1 \\ 1 \\ \sqrt{2} \\ 1 \\ -\sqrt{2} \\ -\sqrt{2} \end{bmatrix}$$

- $w_0 = \frac{1}{8}[-\emptyset(x_1) + \emptyset(x_2) + \emptyset(x_3) - \emptyset(x_4)]$

$$w_0 = \frac{1}{8}\left[-\begin{bmatrix} 1 \\ 1 \\ \sqrt{2} \\ 1 \\ -\sqrt{2} \\ -\sqrt{2} \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ -\sqrt{2} \\ 1 \\ -\sqrt{2} \\ \sqrt{2} \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ -\sqrt{2} \\ 1 \\ \sqrt{2} \\ -\sqrt{2} \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ \sqrt{2} \\ 1 \\ \sqrt{2} \\ \sqrt{2} \end{bmatrix}\right]$$

- $w_0 = \frac{1}{8}\begin{bmatrix} 0 \\ 0 \\ -4\sqrt{2} \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -1/\sqrt{2} \\ 0 \\ 0 \\ 0 \end{bmatrix}$ (the first elements of w_0 indicate bias b=0)

- The optimal hyper plane is, ${w_0}^T \emptyset(x) = 0$

$$\begin{bmatrix} 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \end{bmatrix} = 0$$

Which reduces to $- x_1x_2 = 0$

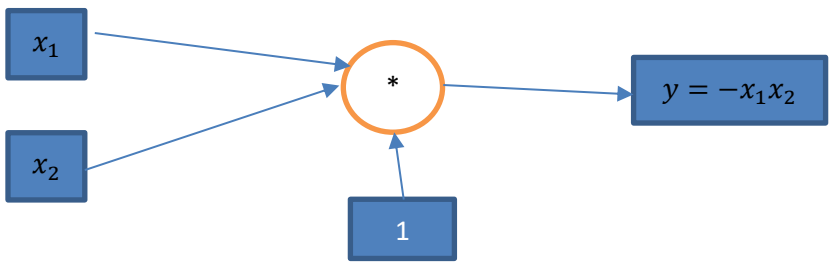


FIGURE4.7 (a): Polynomial Machine for solving the XOR problem

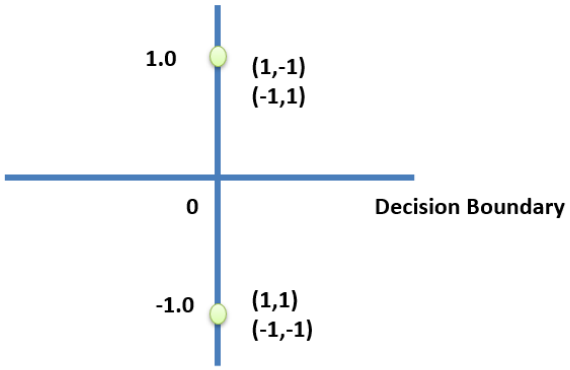


FIGURE 4.7 (b) : Induced image within the feature space because of the four data points of the XOR problem

4.6 Survey on the Feature Extraction:

Wavelet-based feature extraction and selection for classification of installation disturbances using support vector machines by Yakup Demir and Huseyin Eristi in 2010 [38,80]. They gives a replacement approach for the classification of the facility system disturbances using support vector machines (SVMs). The proposed approach is allotted at three serial stages. Firstly, the features to be form the SVM classifier are obtained by using the wavelet transform and a few of various feature extraction techniques. Secondly, the features exposing the only classification accuracy of these features are selected by a feature selection technique called as sequential forward selection. Thirdly, the simplest appropriate input vector for SVM classifier is rummaged. The input vector is started with the primary best feature and incrementally added the chosen features. After the addition of each feature, the performance of the Support vector machine is evaluated. The kernel and penalty parameters of the Support vector machine are determined by cross-validation. The parameter set that provides the littlest misclassification error is retained. Finally, both the noisy and noiseless signals are applied to the classifier given above stages. Experimental results indicate that the proposed classifier is strong and has more high classification accuracy with reference to the opposite approaches within the literature for this problem.

Improved Defect Detection Using Support Vector Machines and Wavelet Feature Extraction supported Vector Quantization and SVD Techniques by D.A. Karras in 2003 [46]. He investigating a completely unique solution to the problem of defect detection from images using the Support Vector Machines. The suggested solution focuses on detecting defects for manufacturing applications from their wavelet transformation and vector determination related properties of the associated wavelet coefficients. More specifically, a completely unique methodology is research for discriminating defects by applying a supervised neural classification technique, namely support vector machine to innovative multidimensional wavelet-based feature vectors. These vectors are extracted from the K-Level 2-D DWT (Discrete Wavelet Transform) transformed original image using Vector Quantization techniques and a Singular Value Decomposition (SVD) Analysis applied to those wavelet domain quantization vectors. The results of the proposed methodology is illustrated in defective textile images where the defective areas are recognized with higher accuracy than the one obtained by applying two rival defect detection methodologies. The primary one of them uses all the wavelet coefficients derived from the k-Level 2-D DWT and involves SVM again within the classification stage, while the opposite uses again all the wavelet coefficients derived from the k-Level 2-D DWT but involves a Multilayer Perceptron neural network within the classification stage. The promising results herein shown outline the importance of judicious selection and processing of 2-D DWT wavelet coefficients for industrial pattern recognition applications also because the generalization performance benefits obtained by involving SVM neural networks rather than other ANN models.

Wavelets and Support Vector Machines for Texture Classification Kashif Mahmood Rajpoot and Nasir Mahmood Rajpoot in 2004 [73]. They present a completely unique texture classification algorithm using 2-D discrete wavelet transform and support vector machines. The DWT is employed to get feature images from individual wavelet sub bands, and an area energy function is computed like each pixel of the feature images. This feature vector is first used for training and afterward for testing the SVM classifier. The experimental setup consists of images from the Brodatz and MIT VisTeX texture databases and a combination of some images therein. The

proposed method produces promising classification results for both single and multiple class texture analysis problems.

A Survey of Feature Selection and have Extraction Techniques in Machine Learning by Samina Khalid, Tehmina Khalil and Shamila Nasreen in 2014 [80]. Within the field of machine learning and pattern recognition, dimensionality reduction is vital area, where many approaches are proposed. Some widely used feature selection and have extraction techniques have analysed with the aim of how effectively these techniques are often wont to achieve high performance of learning algorithms that ultimately improves predictive accuracy of classifier.

Here are the subsequent techniques for the feature extraction [26]: As Suppose there are three sentences:

- THE STUDENT PASS THE TEST .

  Det N V Det N PUNCT

- THE STUDENT WAIT FOR THE PASS .

  Det N V P Det N PUNCT

- TEACHER TEST STUDENTS .

  N V N PUNCT

TABLE 4.3: Emission Probability Matrix

	THE	STUDENT	PASS	TEST	WAIT	FOR	TEACHER	.
Det	$\frac{4}{4}=1$	0	0	0	0	0	0	0
N	$\frac{0}{6}=0$	$\frac{3}{6}=\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{0}{6}=0$	$\frac{0}{6}=0$	$\frac{1}{6}$	$\frac{0}{6}=0$
V	$\frac{0}{3}=0$	$\frac{0}{3}=0$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{0}{3}=0$	$\frac{0}{3}=0$	$\frac{0}{3}=0$
P	$\frac{0}{1}=0$	0	0	0	0	1	0	0
Punct.	$\frac{0}{3}=0$	0	0	0	0	0	0	$\frac{3}{3}=1$

TABLE 4.4: Transition probability Matrix

	Det = T1	N = T2	V = T3	P = T4	PUNCT = T5	Q_F
q_0	2/3	1/3	0	0	0	0
Det	0	1	0	0	0	0
N	0	0	1/2	0	1/2	0
V	1/3	1/3	0	1/3	0	0
P	1	0	0	0	0	0
PUNCT	0	0	0	0	0	1

Step: 1 Input neuron:

Feature vector needs to be generated to train the network. Take the first word of S-1 that is, "THE" and relating Tag is "Det". The feature vector can be built by selecting the first column of Table 4.3. Which covers the feature related to all the possible tag allocated to word “THE”. In a Similar

way selecting the first column and first row of the transition matrix depicted in the Table 4.4. The entries of the first column and first row of transition matrix gives the probabilities of the event of the tag "Det" before the remaining tags and after the remaining tags respectively. Combining all the features for the tag "Det" the feature vector seems like {1, 0, 0, 0, 0, 2/3, 0, 0, 1/3, 1, 0, 2/3, 1/3, 0, 0, 0, 0} .So add up to 17 input neurons will be available in the neural network. Similarly, all the feature vectors for each word shows up in the training set. Total 17 words (designs) including "." are available in the training set.

Step: 2 Hidden neurons:

Total 17 designs are there in the database out of which number of distinct neurons might be considered as Hidden neurons. The training set consists of 7 distinct words. The 7 distinct words considered as focuses are {THE, STUDENT, PASS, TEST, WAIT, FOR, TEACHER,}.

Step: 3 Output neurons:

The objective of the analysis is to get a suitable tag for each word. The input database contains 5 tag viz. {Det, N, V, P, Punct}, let us call it as S = {1, 2, 3, 4, 5}. Typically, only one output neuron is required in the system however it won't coverage.

Step: 4 Simulation process (Testing):

Consider arbitrary part of speech words to test the neural network. Self-assertive test words: {STUDENT WAIT TEST} = {w_1, w_2, w_3}.

1. Consider the word w_2, Find it in the training set. Note its position and the corresponding Tag. The Position is 5 (Table 4.3) and the corresponding tag "V" which is number 3.
2. Construct the feature vector as discussed in step 1 by choosing the 5^{th} column of emission matrix shown in the Table 4.3, 3^{rd} row (ignoring the first row of q_0) of transition matrix depicted in Table 4.4.

3. The feature input vector for w_2 is {0,0,1/3,0,0,0,0,0,0,0,0,1/3,1/3,0,1/3,0,0}. This input vector is fed to the trained network and get to output.
4. The obtained output vector is {0.5009,0.5014,0.89,0.91}
5. Thresholding with 0.55, the final output vector is {0,0,1,1}

Since the experiment is conducted for very small database, the output obtained is here has high threshold (0.55). The accuracy is depending on the distributive large database.

4.7 Comparison of Viterbi and SVM for multiclass Problem:

4.7.1 Training and testing data equal:

In first experiment the training and testing datasets are equal. Two algorithm SVM and Viterbi are tested in python programming platform. Viterbi gives 100 % accuracy and SVM gives 59 % accuracy.

TABLE: 4.5: Classification of Same datasets

Viterbi Analysis	SVM Analysis
Number of of Training data: 569 words	Number of of Training data : 569 words
Number of of Testing data : 569 words	Number of of Testing data : 569 words , Number of of Tag: 35
Number of of Tag: 35	Input Neuron: 105 Hidden Neuron : 325 (Distinct word).Output Neuron : 35
Accuracy : 100%	Accuracy : 59%

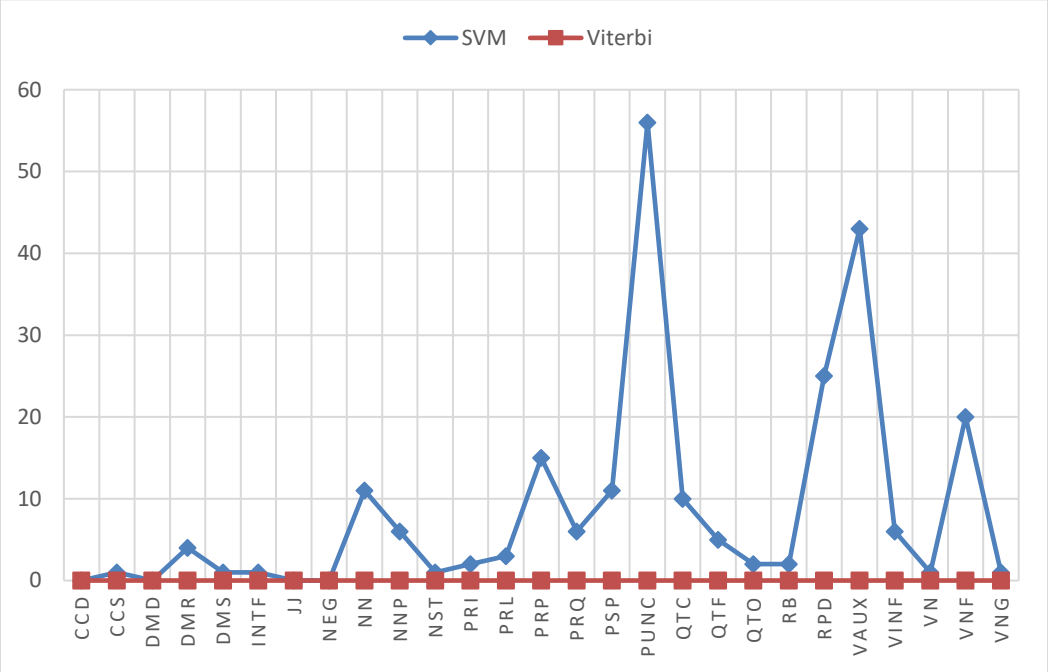


FIGURE 4.8 : Graphical Representation of the Table 4.5

4.7.2 Training and Testing data are Reshuffle:

In second experiment, 10 to 12 sentences are testing dataset viterbi and SVM algorithm are used in python programming platform. Viterbi has given 100 % accuracy and SVM gives 42 % accuracy.

TABLE 4.6: Classification for Reshuffled data

Viterbi Analysis	SVM Analysis
Number of of Training data : 569 words	Number of of Training data : 569 words
Number of of testing Data :12 sentences	Number of of Testing data : 569 words , Number of of Tag: 35
Number of of Tag: 35	Input Neuron : 105 ,Hidden Neuron : 325(Distinct word), Output Neuron : 35
Accuracy : 100%	Accuracy : 42%

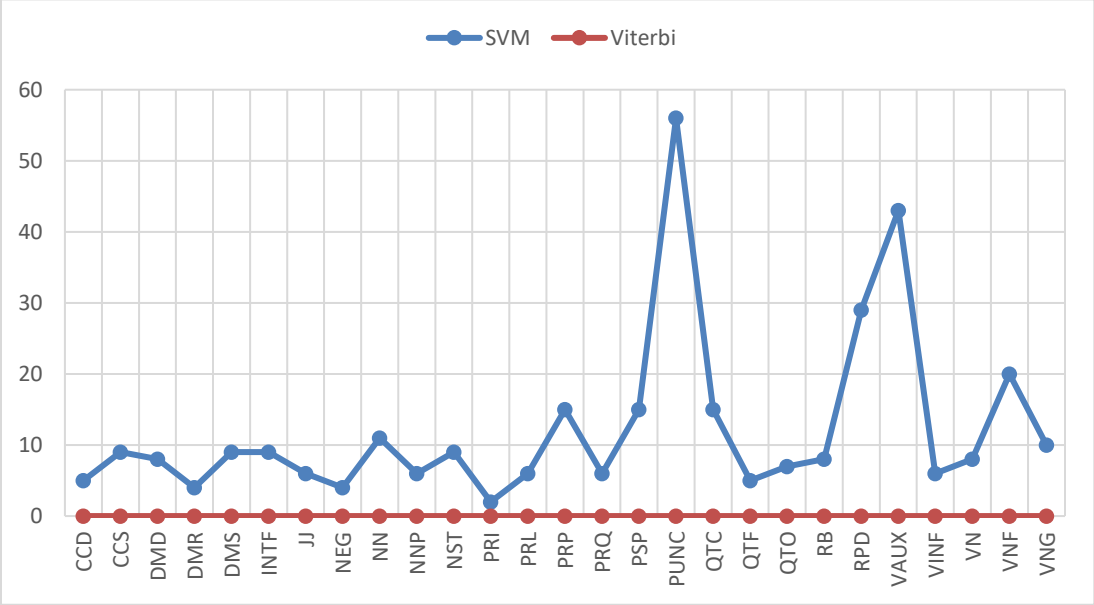


FIGURE 4.9 : Graphical Representation of the Table 4.6

4.7.3 Number of word available in training data set (not in testing)

Third experiment some words are changing in such sentences which are not in testing dataset. Python programming used for both algorithm like Viterbi and SVM. Viterbi gives 75 % accuracy and SVM gives 36.8% accuracy.

TABLE 4.7: Classification of self-generated sentences from the database

Viterbi Analysis	**SVM Analysis**
Number of of Training data : 569	Number of of Training data : 569
Number of of testing Data :10 sentences	Number of of Testing data : 150 words, Number of of Tag: 35
Number of of Tag: 35	Input Neuron : 105 ,Hidden Neuron : 50(Distinct word), Output Neuron : 35
Accuracy : 70 %	Accuracy : 36.8 %

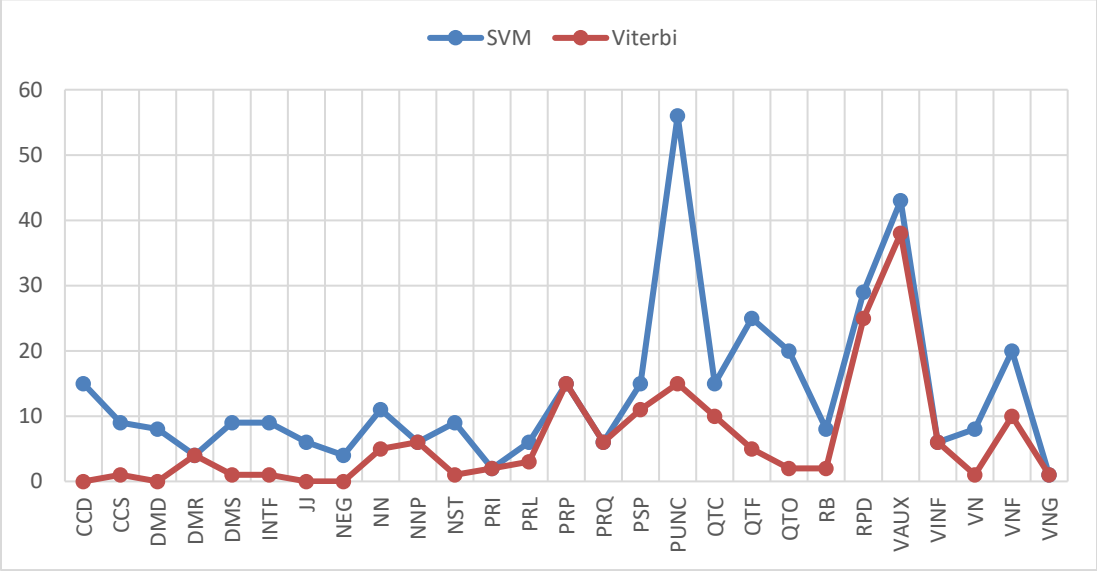


FIGURE 4.10 : Graphical Representation of the Table 4.7

4.7.4 Number of words not available in training and testing data set (unknown words)

Viterbi and SVM algorithm are used in python programming language. In fourth experiment taking such sentences which are not in training and testing datasets (unknown words) Viterbi gives 80 % and SVM gives 32.6 % accuracy.

TABLE 4.8 : Classification of self-generated sentences with some external words

Viterbi Analysis	**SVM Analysis**
Number of of Training data: 569	Number of of Training data: 569
Number of of testing Data :10 sentences	Number of of Testing data: 102 words, Number of of Tag: 35
Number of of Tag: 35	Input Neuron: 105, Hidden Neuron: 102(Distinct word), Output Neuron : 35
Accuracy: 80 %	Accuracy: 32.6 %

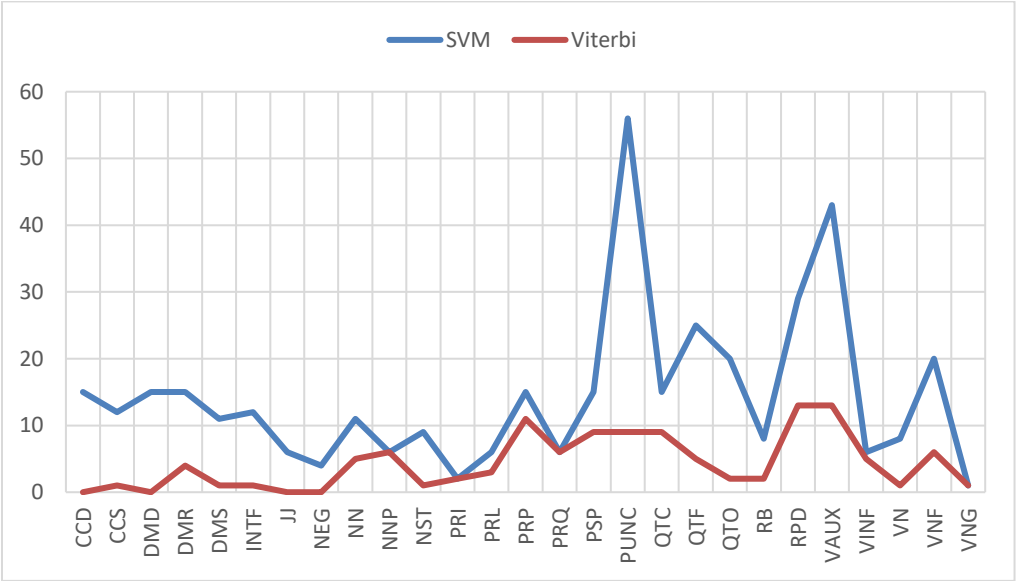


FIGURE 4.11 : Graphical Representation of the Table 4.8

4.7.5 Number of confusing words (i.e, tagging issue- possibility of more than one tag of each word

Fifth experiment are different from the above four experiment. Same algorithm and language will be applicable. Using such sentences which are not training and testing dataset moreover each word have multiple tag. Viterbi gives 52.57% accuracy where as SVM gives 27 % accuracy.

TABLE 4.9 : Classification of confusing words

Viterbi Analysis	**SVM Analysis**
Number of of Training data : 569	Number of of Training data : 569
Number of of testing Data :8 sentences (51 words)	Number of of Testing data : 51 words, Number of Tag: 35
Number of of Tag: 35	Input Neuron : 41 ,Hidden Neuron : 32(Distinct word), Output Neuron : 13
Accuracy: 52.57%	Accuracy: 27 %

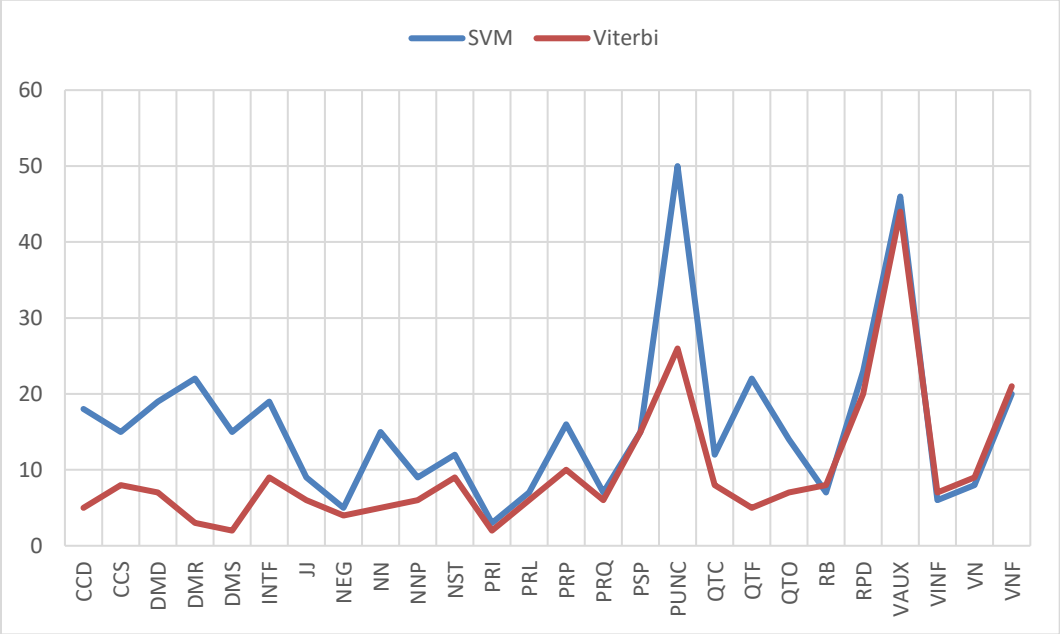


FIGURE 4.12 : Graphical Representation of the Table 4.9

```
# -*- coding: utf-8 -*-
"""
Created on Wed Apr  4 14:36:23 2018

@author: SONU96
"""

import svmtempdata
X = svmtempdata.X
Y = svmtempdata.y

#lis2 = []
#for t in X:
#     lis = ''
#     li = []
#     for v in t:
#          lis += str(v)
#     li.append(int(lis))
#     lis2.append(li)
#
#print(lis2)
#
#ylist = []
#for y in Y:
#     print(y)
#     ylist.append(y)

from sklearn import svm

clf = svm.SVC(kernel='rbf')
clf.fit(X, Y)
i=0
cnt=0
for t in X:
     print(t[0])
     predicted = clf.predict(t[0])
     print("Predicated ",predicted[0] ," Original ", Y[i])
     if(predicted[0]==Y[i]):
          cnt=cnt+1
     i=i+1;
```

FIGURE 4.13 (a) : Python programming for SVM

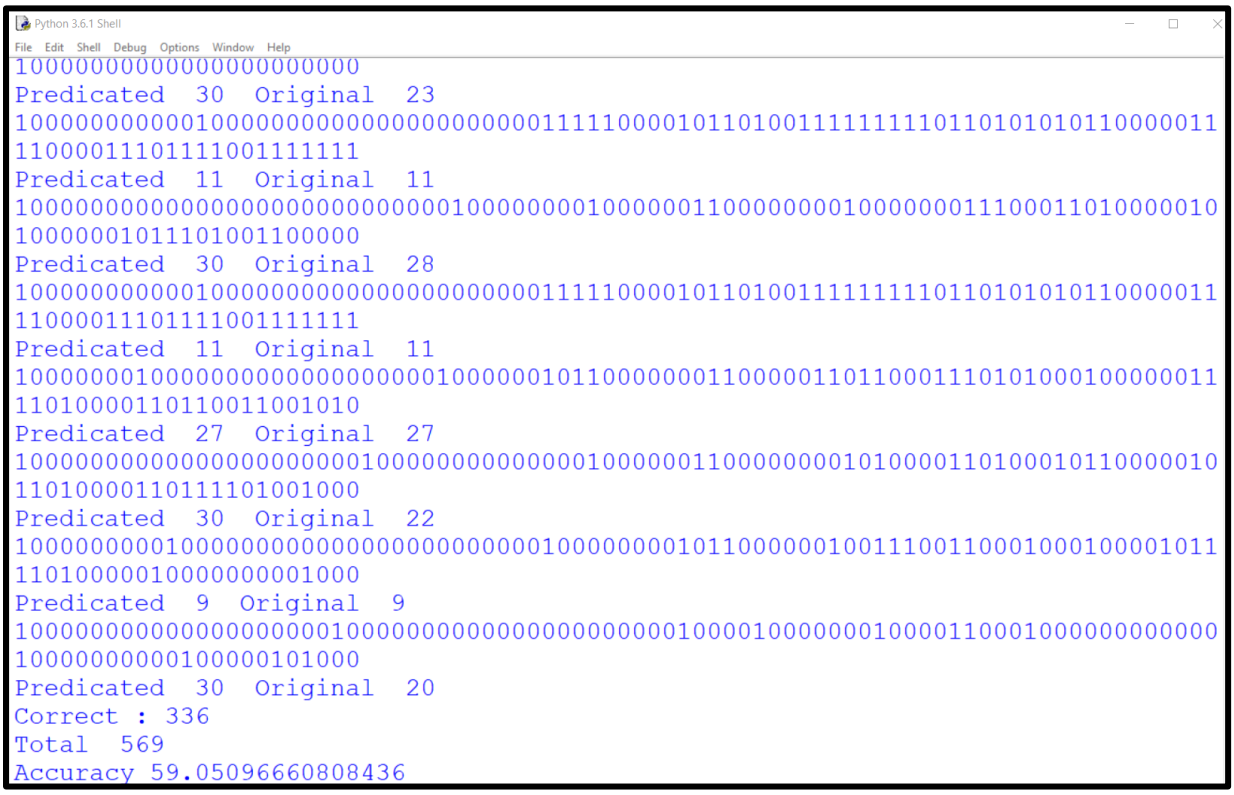


FIGURE 4.13 (b) : Implmentation of SVM

FIGURE 4.14 (a): Python programming for Viterbi

```
['STAART', 'હું', 'તરી', 'શકું', 'છુ', '.']
Total Test Data 6
Distinct Tags
{'VNF', 'XAUX', 'QTF', 'DMD', 'RPD', 'VAUX', 'INTF', 'NN', 'PRP', 'DMS', 'PRI', 'DMR', 'PRL', 'JJ', 'END', 'PUNC',
'NST', 'VM', 'VINF', 'START', 'CCS', 'PSP', 'PRQ', 'NEG', 'VN', 'RB', 'NNP', 'CCD', 'VNG', 'QTO', 'QTC'}
Count :31
Tags for Word હું are ['PRP']
Prediction for Word No :1 ->હું Actual ['PRP'] Predicated PRP
Tags for Word તરી are []
No Word Founded તરી
Prediction for Word No :2 ->તરી Actual [] Predicated QTC
Tags for Word શકું are []
No Word Founded શકું
Prediction for Word No :3 ->શકું Actual [] Predicated QTC
Tags for Word છુ are []
No Word Founded છુ
Prediction for Word No :4 ->છુ Actual [] Predicated QTC
Tags for Word. are ['PUNC']
Prediction for Word No :5 ->. Actual ['PUNC'] Predicated PUNC
Total 5
Correct 2
Accuracy 40
```

FIGURE 4.14 (b): Implementation of Viterbi

4.8 Wavelet Based Neural Network

4.8.1 Wavelet Theory

Firstly, proposed by Grossman and Morlet in the 1980s, wavelet theory [9] is a mathematical theory and analysis method to make up the shortages of Fourier transform. In the field of signal processing, the most widely used analysis method is the Fourier transform, but it has obvious deficiency that the Fourier transform has no distinguishable ability in the time domain, because the time information is not included in the results of Fourier transform. Wavelet is special waveform with the mean 0 and the limited length.

Wavelet function is constructed through a series of basic transformation with a mother wavelet function. Not all functions can be used as wavelet mother function if a wavelet function is to be available and then develop into a good wavelet transform function, it must satisfy many conditions. Therefore, it is difficult to find the practical wavelet function. In the practical wavelet functions, some of them do not have expressions.

Let $\varphi(t)$ be a square integrable function, that is $\varphi(t) \in L^2(R)$. If its Fourier transform $\Psi(w)$ can satisfy the following compatibility condition:

$$\int_R \frac{|\psi(w)|^2}{w} dw < \infty \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4.17)$$

then $\varphi(t)$ is called a basic wavelet or mother wavelet function. We make translation and scale for wavelet function, the translation factor τ, and the scale factor (also known as the expansion factor) a , so that we get function :

$$\psi_{a,\tau}(t) = \frac{1}{\sqrt{|a|}}\psi(\frac{t-\tau}{a}) \quad a > 0\,, \tau \in R \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..(4.18)$$

As the translation factor τ and the scale factor a are continuous variables, their value can be positive or negative; so $\psi_{a,\tau}(w)$ is called continuous wavelet function (also called the mother wavelet function).

Wavelet transform calculates the inner product between the signal $x(t)$ with mother wavelet function

$$f_x(a,\tau) = \frac{1}{\sqrt{a}}\int_{-\infty}^{\infty} x(t)\varphi^*(\frac{t-\tau}{a})dt \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..(4.19)$$

Equivalent expression in time domain is given as

$$f_x(a,\tau) = \frac{\sqrt{a}}{2\pi}\int_{-\infty}^{\infty} X(w)\Psi^*(aw)e^{jw\tau}dw \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4.20)$$

where $a > 0\,, \tau \in R$ $X(w)$and $\Psi(w)$ are the Fourier transform of $x(t)$ and $\varphi(t)$ respectively.

The conclusion can be drawn from (3) and (4) that is wavelet analysis can analyze the local characteristics of the signal through the mother wavelet function transformation; therefore, wavelet theory is considered to be the breakthrough for the Fourier transform, and the theory has been

successfully applied to image processing, optical devices detection, and signal analysis and other fields.

4.8.2 Wavelet Neural Network

Wavelet transform has time-frequency localization property and focal features and neural network (NN) has self-adaptive, fault tolerance, robustness, and strong inference ability. How to combine the advantages of wavelet transform and NN to solve practical problems has been one of the hot spots. So-called wavelet neural network (WNN) or wavelet network (WN) is a variety of two techniques and inherits the advantages of the neural network and wavelet transformation. WNN uses the wavelet function as the activation function instead of the Sigmoid activation function.
For WNN, its topology is based on BP network; the transfer function of hidden layer nodes is the mother wavelet function; and the network signal is prior to transmission while error is backpropagation in the training process. The network topology is shown in 4.12

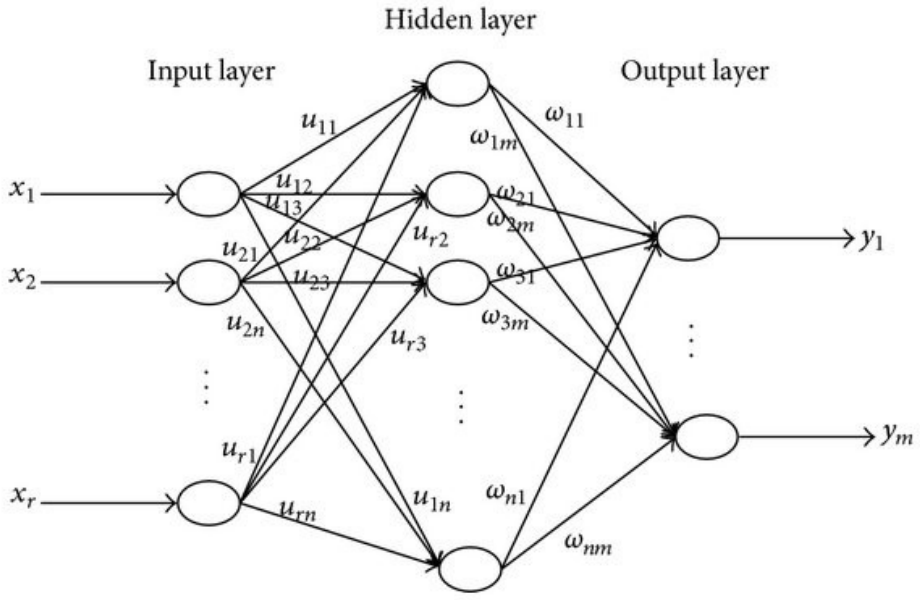


FIGURE 4.15: Architecture of wavelet neural network

In Figure 4.15 $x_1, x_2, x_3, \ldots\ldots\ldots\ldots x_n$ is the input vector; $y_1, y_2, y_3, \ldots\ldots\ldots\ldots y_l$ is the predicted output; w_{ij} and w_{kj} are the weights connecting every layer; and h_j is mother wavelet function.

For the input signal sequence $x = (x_1, x_2, x_3, \ldots\ldots\ldots\ldots x_n)$ he output of the hidden layer is calculated as

$$h(j) = h_j\left[\frac{\sum_{i=1}^{n} w_{ij}x_i - b_j}{a_j}\right] \quad j = 1,2,\ldots\ldots\ldots m \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4.21)$$

where $h(j)$ is output value for the node j ; in the hidden layer; h_j is the mother wavelet function; w_{ij} is weight connecting the input layer and hidden layer; b_j is the shift factor, and a_j is the stretch factor for h_j.

Currently, the choice of mother wavelet functions has not yet formed a standard theory; commonly used wavelet functions are Morlet, Haar, Daubechies (dbN), Symlet (symN), Meryer, Coiflet, Biorthogonal wavelets, and so on.

The output of the output layer is calculated as

$$y(k) = \sum_{i=1}^{m} w_{ik}h_i \,, k = 1,2,\ldots\ldots\ldots l \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4.22)$$

Where h_i is the output value for node i in the hidden layer; w_{ik} is weight connecting the hidden layer and output layer; l and m are the number of nodes for output layer and the hidden layer, respectively.

For WNN, the updating weight algorithm is similar to back propagation (BP) neural network the gradient method is used to update mother wavelet function parameters and connection weights between the layers, making the prediction output closer and closer to the desired output. The weights of WNN and the parameters of wavelet function are updated as follows.

(1) Calculating the prediction error of WNN

$$e = \sum_{k=1}^{m} y_n(k) - y(k) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4.23)$$

where, $y(k)$ is the predicted output value $y_n(k)$ is the expected output value for the network.

(2) Updating the weights of WNN and the parameters of wavelet function according to the prediction error

$$w_{n,k}^{(i+1)} = w_{n,k}^{(i)} + \Delta w_{n,k}^{(i+1)},$$

$$a_k^{(i+1)} = a_k^{(i)} + \Delta a_k^{(i+1)}, \qquad (4.24)$$

$$b_k^{(i+1)} = b_k^{(i)} + \Delta b_k^{(i+1)}$$

Where $\Delta w_{n,k}^{(i+1)}$, $\Delta a_k^{(i+1)}$ and $\Delta b_k^{(i+1)}$ are calculated by the network prediction error:

$$\Delta w_{n,k}^{(i+1)} = -\eta \frac{\partial e}{\partial w_{n,k}^{(i)}}$$

$$\Delta a_k^{(i+1)} = -\eta \frac{\partial e}{\partial a_k^{(i)}} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.25)$$

$$\Delta b_k^{(i+1)} = -\eta \frac{\partial e}{\partial b_k^{(i)}}$$

where η is the learning rate.

The process of training WNN is as follows

(1) Data pre-processing: first, the original data is quantified and normalized, and then the data is divided into training set and testing set for network training and testing, respectively.

(2) Initializing WNN: connection weights w_{ij} and w_{jk} translation factor b_k and scale factor a_k are randomly initialized, and the learning rate η is set.

(3) Training network: input the training set into WNN, compute network predicted output values, and calculate the error e between output and the expected value.

(4) Updating the weights: update mother wavelet function parameters and network weights according to the prediction error e , making the predictive value of the network as close to actual values.

(5) If the results satisfy the given conditions, use the testing set to test the network, otherwise, return to Step 3.

4.8.2.1 Creating the Mother Wavelet Function Library

Because many mother wavelet functions have no specific expression, we cannot work out their derivatives. Following are mother wavelet functions to build a library wavefunciton. Their expressions are as follows:

(1) Haar wavelet function:

$$\Psi(t) = \begin{cases} 1\ , & 0 \le t \le \frac{1}{2} \\ -1, & \frac{1}{2} \le t \le 1 \\ 0, & other \end{cases}$$

(2) Gaussian wavelet function:

$$\Psi(t) = \frac{t}{\sqrt{2\pi}} exp\left(-\frac{t^2}{2}\right)$$

(3) Morlet wavelet function:

$$\Psi(t) = \cos(1.75t) exp\left(-\frac{t^2}{2}\right)$$

4.8.3 Discreet Wavelet Transform:

In DWT, a signal is analysed with a small number of scales with varying number of translations at each scale. A critical sampling of the CWT $W(a,b)$ is obtained by substituting $a = 2^{-j}$and $b = k2^{-j}$where j and k are integers representing the scale and translation. Upon this substitution,

$$\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k)$$

These wavelets for all integers j and k produce an orthonormal basis. $\psi_{0,0}$ (t) =ψ(t) is the mother wavelet. Other wavelets are produced by translation and dilation of the mother wavelet. Discrete wavelet transforms, denoted by $W(j,k)$, is hence given by

$$\psi_{j,k}(t) = \int f(t)2^{j/2}\psi(2^j t - k)\, dt$$

$$m, n \in Z \text{ and for } f \in L^2(R)$$

The term critical sampling is used to ensure that a minimum number of wavelet coefficients are retained to represent all the information present in the original function. In CWT, transform coefficients are found for every (a,b) combination, where as in DWT, transform coefficients are found only at very few points.

4.8.4 Daubechies Discrete Wavelets:

The three properties that a function must satisfy to be classed as a wavelet. For a wavelet, and its scaling function, to be useful in the DWT, there are more conditions that must be satisfied:

$$\sum_k h_k = \sqrt{2}$$

$$\sum\nolimits_k (-1)^k k^m h_k = 0 \qquad for\ m = 0,1,2 \dots \dots \dots \frac{N}{2} - 1.$$

$$\sum_k h_k h_{k+2m} = \begin{cases} 0 & , \quad for\ \ m = 0,1,2 \dots \dots \dots \frac{N}{2} - 1 \\ 1 & , \quad for \quad m = 0 \end{cases}$$

Ingrid Daubechies discovered a category of wavelets, which are characterized by orthonormal basis functions [35]. That is, the mother wavelet is orthonormal to every function obtained by shifting it by multiples of 2^j and dilating it by an element of 2^j (where j ∈ Z). The Haar wavelet was a two-term member of the category of discrete Daubechies wavelets. we will easily make sure the above conditions are satisfied for the Haar wavelet, remembering that $h_0=h_1=1/\sqrt{2}$. The Daubechies D(4) wavelet may be a four-term member of an equivalent class. The four scaling function coefficients, which solve the above equation for N = 4, are specified as follows:

$$h_0 = \frac{1+\sqrt{3}}{4\sqrt{2}} \quad , \quad h_1 = \frac{3+\sqrt{3}}{4\sqrt{2}} \quad , \; h_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}, h_3 = \frac{1-\sqrt{3}}{4\sqrt{2}} \quad .$$

The scaling function for the D(4) wavelet can be built up recursively from these coefficients. The wavelet function can be built from the coefficients $g_k = (-1)^k h_{4-k-1}$ .The Daubechies orthonormal wavelets of up to 20 coefficients (even numbered only) are commonly used in wavelet analysis.

4.9 Wavelet Feature Extraction

The survey of Part of Speech Tagging for Indian languages is covered [66,9].. Out of which 1140 samples (patterns) are used for training and the remaining 569 samples for testing. The database is distributed in to n = 35 tags. Network architecture consists of 35 x 3=105 input neurons, 1140 hidden neurons which plays a role of centers C_i(i = 1, 2, …, 569) shown in (4.6) and 35 neurons in output layer. Transition$(T)_{n\times n}$ and Emission probability matrices $(E)_{n\times m}$ are constructed for both the sets viz. training and testing. Transition matrix demonstrates the probability of occurrence of one tag (state) after another tag (state) hence becomes a square matrix 35 x 35. Whereas the emission matrix is that the matrix of probability distribution of each Gujarati word is allotted the respective tag hence it is of the size $n \times m$ (number of Gujarati words). In order to fetch the features for i^{th}word say x_i , the i^{th}row,i^{th}column of the transition matrix and i^{th}row of the emission matrix are combined hence becomes $35 \times 3 = 105$ features for each word. Therefore,

the wavelet based neural network consists of 105 input neurons all the patterns (or Gujarati words) are used as a centre. Euclidean distance is calculated between patterns and centre. As there are 35 tags, 35 output neurons constitute the output layer of the network. as an example, if the word belongs to NN (common noun) category which is that the first tag of the tag set then the primary neuron features a value 1 and every one others are 0.

4.9.1 Result Analysis:

The training database contains 1140 words whereas testing set consists of 569 words. Both databases do not contain the words with multiple tags i.e., same Gujarati word with more than one different tag. Experiments demonstrate that all the architectures are better able to identify the suitable tags as far as the training set is concern but in the case of testing set except WNN other network performs efficiently. The input layer in the case of SVM consists of 105 neurons. In order to employ wavelet-based features in the input layer 23 zeros are (add) padded in the input vector for each pattern, so it becomes 128 (27). This is how, to get the discrete signal of the resolution m = 7. By applying Daubechies D4 wavelet transform for one level, we obtained 64 (26) wavelet coefficients as the features of the input layer. By carry out the same number of hidden neurons and the output neurons in the experiments shown in 4.6, we got poor tagging accuracies less than 25% in each case and hence the experiment failed to capture significant features from the words.

4.10 Summary:

In this POS tagging approach is introduced for the Gujarati Text. Two techniques are employed viz. SVM and Viterbi algorithm for this purpose. Section 4.7 demonstrates the outcome of the experiment on two data sets viz. training (569 words) and testing (569 words). Transition and emission probability matrices are constructed for both the techniques. Features are extracted from these matrices and used as an Input layer for SVM as shown in section 4.6. Fully connected i.e. all the patterns are taken as centers SVM architecture is trained using training set and outputs are validated on both training and testing sets. The result is compared with the traditional statistics-

based Viterbi algorithm. Table 4.5 shows that both the approaches yields satisfactory accuracy more than 96% as far as the training samples are concern but SVM gives with 59 % accuracy. Fig 4.5 represent the all tag for SVM in how much time wrong. For illustration NN aren't right 10 times, likewise PUNC (punctuation) aren't right 56 times etc.... similarly for the Fig 4.6 represent the all tag for SVM in VAUX (verb auxiliary) aren't right 43 times etc…similarly for Fig 4.7 represent the all tag for SVM ,VAUX aren't right 42 times and Viterbi in VAUX aren't right 39 times and for SVM , PUNC (punctuation) aren't right 58 times etc… for Viterbi , PUNC (punctuation) aren't right 15 times etc… Fig 4.8 represent the all tag for SVM VAUX aren't right 42 times and Viterbi in VAUX aren't right 12 times and for SVM ,PUNC (punctuation) aren't right 58 times etc… Viterbi , PUNC (punctuation) aren't right 9 times etc… and The confusion matrix (Table 4.9) is generated on the output of SVM on testing set. The accuracy may further be improved by collecting uniformly distributed data set.

In Comparison of SVM and wavelet based neural network, Words of testing set with less frequency of occurrence may be 1 or 2 in training set are failed to get an appropriate tag in the case of SVM architecture, it concludes that SVM technique which is far better accuracy as compared to the opposite ANN architectures which require training accuracy is extremely poor.

CHAPTER – 5

Insights into Non-projectivity in Gujarati

5.1 Overview of Gujarati Language Analyzer:

Gujarati Morph analyzer has been developed employing a hybrid approach that combines statistical, knowledge based and paradigm-based approach [103]. For Paradigm construction, sample corpus of inflected words is taken. A listing is prepared for all possible suffixes from the sample data. The words which take similar set of suffixes are grouped into single paradigm. For all the words be-longing to same paradigm, the rule to from root word and set of possible suffixes to generate other word forms remain same. Gujarati nouns inflect in gender, number and case. Gujarati has three genders and two numbers. Adjectives are often classified into variant and non-variant adjectives based on the inflections that they take. A non-variant adjective do not inflect with gender. Gujarati verb inflect in gender, number, case and tense. Verbs and adjectives require agreement with Noun. Each paradigm consists of rules to obtain root word, various inflections possible with that paradigm and a representing word for that paradigm. An inflected word is given as an input to the framework. The framework checks suffix and determines one or more matching paradigm for input word. It is possible that an equivalent rule is present in more than one paradigm. For every matched paradigm, the system applies rule to generate root word and provides root word as output. It is possible that single inflected word is mapped with more than one root word. .So to eliminate this multiple output two different methods viz. knowledge based and statistical method are incorporated.

5.1.1 Types of morphology:

1. Inflectional Morphology: Modification of a word to express different grammatical categories. Inflection marks categories such as person, tense, and case; *e.g.,* "sings" contains a final *-s,* marker of the 3rd person singular. For Example: Cats, men, etc.
2. Derivational Morphology: Creation of a replacement word from existing word by changing grammatical category. Derivation is the formation of new words from existing words; *e.g.,* "singer" from "sing" and "acceptable" from "accept." Example: happiness, brotherhood etc.

5.1.2 Approaches to Morphology:

There are three principal approaches to morphology.

1. Morpheme based morphology: Word forms are analyzed as arrangements of morphemes. A morpheme may be a smallest language unit with a grammatical function.
2. Lexeme Based Morphology: It always takes what's called an "item-and-process" approach. Rather than analyzing a word form as a set of morphemes arranges in a sequence, a word forms said to be the results of applying rules that alter a word-form or stem in order to produce a new one.
3. Word Based Morphology: It is usually a word -and-paradigm approach. Instead of stating rules to combine morphemes into word forms, or to generate word forms from stem, word-based morphology states generalization that hold between the forms of inflectional paradigms.

5.2 Introduction:

Gujarati, being a morphologically rich, flexible word order language, brings challenges like handling non-projectivity in parsing [51]. During this work, working at non-projectivity for Gujarati. Non-projectivity has been analyzed from two perspectives: graph properties that restrict non-projectivity and linguistic phenomenon behind non-projectivity. Here checked out graph constraints like planarity, gap degree, edge degree and well-nestedness on structures. The analyze of non-projectivity in Gujarati in terms of various linguistic parameters like the causes of non-projectivity, its rigidity (possibility of reordering) and whether the reordered construction is that

the natural one. Non-projectivity occurs when dependents don't either immediately follow or precede their heads in a sentence. These dependents could also be opened up over a discontinuous region of the sentence. It is well-known that this poses problems for both theoretical grammar formalisms as well as parsing systems. Gujarati may be a verb final, flexible ordering language and thus, has frequent occurrences of non-projectivity in its dependency structures. It shows that major chunk of errors in their parser is due to non-projectivity. So, there's a requirement to research non-projectivity in Gujarati for a far better insight into such constructions. It would like to say here, that as far as aware, there hasn't been any plan to study non-projectivity in Gujarati before this work. This work is a step forward in this direction. Non-projectivity are often analyzed from two aspects. a) In terms of graph properties which restrict non-projectivity and b) in terms of linguistic phenomenon giving rise to non-projectivity. While a) gives a thought of the type of grammar formalisms and parsing algorithms required to handle non-projective cases in a language, b) gives an insight into the linguistic cues necessary to identity non-projective sentences in a language.

Parsing systems can explore algorithms and make approximations supported the coverage of those graph properties on the tree bank and linguistic cues are often used as features to limit the generation of non-projective constructions. Similarly, the analyses based on these aspects also can be used to come up with broad coverage grammar formalisms for the language. Graph constraints like projectivity, planarity, gap degree, edge degree and well-nestedness are utilized in previous works to seem at non-projective constructions in tree banks like PDT and DDT. Employ these constraints in our work too. Apart from these graph constraints, it would be also check out non-projective constructions in terms of various parameters like factors resulting in non-projectivity, its rigidity, its approximate projective construction and whether it's the natural one. The dependency relations in the Treebank are syntactico-semantic in nature where the main verb is that the central binding element of the sentence. The arguments including the adjuncts are annotated taking the meaning of the verb into consideration. The participants in an action are labelled with karaka relations. Syntactic cues like case-endings and markers like post-positions and verbal inflections help in identifying appropriate karakas. The dependency tagset in the annotation scheme has 28 relations in it. These include six basic karaka relations (adhikarana [location],

apaadaan [source], sampradaan [recipient], karana [instrument], karma [theme] and karta [agent]. The remainder of the labels are non-karaka labels like vmod, adv, nmod, rbmod, jjmod., etc…The tagset also includes special labels like pof and ccof, which aren't dependency relations in the strict sense. They are used to handle special constructions like conjunct verbs (ex: - પ્રશ્ન પૂછ્યો (question did)), coordinating conjunctions and ellipses. In the annotation scheme relations are marked between chunks instead of words. A chunk (with boundaries marked) represents a group of adjacent words which are in dependency relation with each other, and are connected to the rest of the words by single incoming dependency arc. The relations among the words during a chunk aren't marked. Thus, during a dependency, each node may be a chunk and therefore the edge represents the relations between the connected nodes labelled with the karaka or other relations. All the modifier-modified relations between the heads of the chunks (inter-chunk relations) are marked during this manner. To get the complete dependency tree, used to an automatic rule based intra-chunk relation identifier.

5.3 Non projectivity and graph property:

In this section, dependency graph formally and discuss standard properties such as single headedness, acyclicity and projectivity [51]. Look at complex graph constraints like gap degree, edge degree, planarity and well-nestedness which can be used to restrict non-projectivity in graphs. In what follows, a dependency graph

for an input sequence of words $w_1, w_2, \dots w_n$is an unlabeled directed graph

$D = (W, Z)$ Where $W$is a set of nodes and $Z\ is$ a set of directed edges on these nodes.

$w_i \rightarrow w_j$ denotes an edge from w_i to w_j , $(w_i, w_j) \in Z$. $\rightarrow *$ is used to denote the reflexive and transitive closure of the relation.

$w_i \rightarrow * w_j$ means that the node w_i dominates the node w_j , i.e., there is a (possibly empty) path from w_i to w_j.

$w_i \leftrightarrow w_j$ denotes an edge from w_i to w_j or vice versa.

For a given node w_i , the set of nodes dominated by $w_i$is the projection of w_i .Here use $\pi(w_i)$to refer to the projection of w_i arranged in ascending order. Every dependency graph satisfies two constraints: acyclicity and single head. Acyclicity refers to there being no cycles in the graph. Single head refers to each node in the graph D having exactly one incoming edge (except the one which is at the root). While acyclicity and single head constraints are satisfied by dependency graphs in almost all dependency theories. Projectivity is a stricter constraint used and helps in reducing parsing complexities.

5.3.1 Projectivity:

If node $w_k$depends on node w_i , then all nodes between w_i and $w_k$are also subordinate to w_i(i.e dominated by w_i)

$$w_i \rightarrow w_k \Rightarrow w_i \rightarrow* w_j$$

$$\forall\, w_j \in W : (w_i < w_j < w_k \vee w_i > w_j > w_k)$$

Any graph which doesn't satisfy this constraint is non-projective. Unlike acyclicity and the single head constraints, which impose restrictions on the dependency relation in and of itself, projectivity constrains the communication between the dependency relations and therefore the order of the nodes in the sentence. Graph properties like planarity, gap degree, edge degree and well-nestedness have been proposed in the literature to constrain grammar formalisms and parsing algorithms from looking at unrestricted non-projectivity.

5.3.2 Planarity:

A dependency graph is planar if edges do not cross when drawn above the sentence [14]. It is similar to projectivity except that the arc from dummy node at the beginning (or the end) to the root node is not considered.

$$\forall\, (w_i, w_j, w_k, w_l) \in W$$

$$\neg\, (\, (w_i \leftrightarrow w_k \wedge w_j \leftrightarrow w_l) \wedge (w_i < w_j < w_k < w_l)\,)$$

5.4 Cases of non-projectivity:

To study of the instances of non-projectivity. Classify these instances based on factors leading to non-projectivity and present our analysis of them. For each of these classes, the rigidity of these non-projective constructions and their best projective approximation possible by reordering. Rigidity here is that the reorder ability of the constructions retaining the gross meaning. Gross meaning refers to the meaning of the sentence not taking the discourse and topic-focus into consideration, which is how parsing is usually done. Study of rigidity is important from natural language generation perspective. Sentence generation from projective structures is simpler and more efficient than from non-projective ones. Non-projectivity in constructions that are non-rigid are often effectively addressed through projectivization. Further, if these approximations are more natural compared to the non-projective ones as this impacts sentence generation quality. A natural construction is that the one most preferred by native speakers of that language. Also, it more or less abides by the well-established rules and patterns of the language. The non-projectivity is caused in Gujarati, because of various linguistic phenomena manifested in the language, like relative co-relative constructions, paired connectives, complex coordinating structures, interventions in verbal arguments by non-verbal modifiers, shared arguments in non-finite clauses, movement of modifiers, ellipsis etc. Also, non-projectivity in Gujarati can occur within a clause (intra-clausal) also as between elements across clauses (inter-clausal) [51].

5.4.1 Relative co-relative constructions:

The pattern in co-relatives is that a demonstrative pronoun, which also functions as determiner in Gujarati, such as તે (that) always occurs in correlation with a relative pronoun, જે (which)In fact, the language employs a series of such pronouns: e.g., જે-તે (which-that), જ્યાં-ત્યાં (where-there), જ્યારે-ત્યારે (when-then). Non-projectivity is seen to occur in relative correlative constructions

with pairs such as જ્યાં-ત્યાં. if the clause beginning with the ત્યાં precedes the જ્યાં clause as seen in Fig. 5.1.

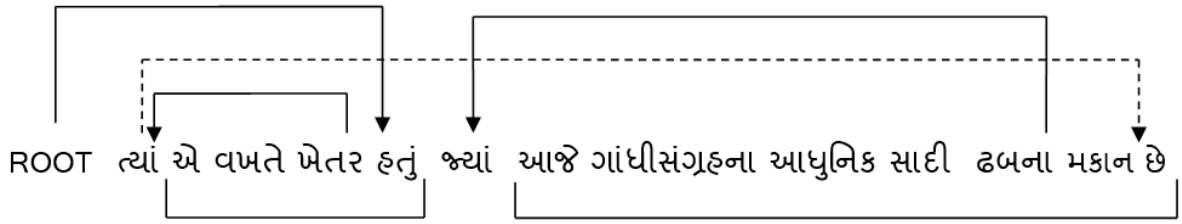


FIGURE 5.1: Relative co-relative construction

There was a farm at the time where today there is a simple modern house of Gandhi Collection. ત્યાં (w_1) એવખતે (w_2) ખેતર (w_3) હતું (w_4) જ્યાં (w_5) આજે (w_6) ગાંધીસંગ્રહના (w_7) આધુનિક (w_8) સાદી ઢબના (w_9) મકાન (w_{10}) છે. (w_{11}) From the section 5.3.1 the definition projectivity suppose

$$w_1 \rightarrow w_{11} \Rightarrow w_1 \rightarrow * w_4$$
$$w_i \in w \;\; i = 1,2,3 \ldots\ldots 11, \; (w_1 < w_4 < w_{11} \;\; \vee w_1 > w_4 > w_{11})$$

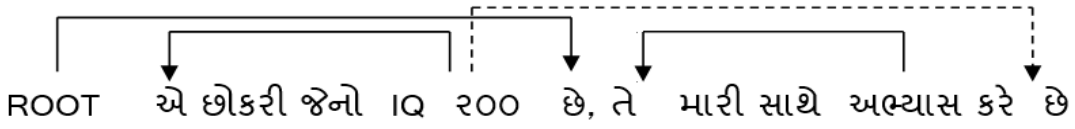
Now from the Fig.5.1 it shows that i.e.; $w_1 \rightarrow * w_4$ $(i.e., w_1 depends\ on\ w_4)$ and also $w_1 \rightarrow * w_{11} (i.e., w_1 depends\ on\ w_{11})$ Therefore, it is non-projectivity.

If the clause with the relative pronoun comes before the clause with the demonstrative pronoun, non-projectivity can be ruled out. So, this class of non-projective constructions is not rigid since projective structures can be obtained by reordering without any loss of meaning. The projective case is relatively more natural than the non-projective one. This is reaffirmed in the corpus where the projective relative co-relative structures are more frequent than the non-projective sentences. In the example in Fig.5.1, the sentence can be reordered by moving the ત્યાં clause to the right of the જ્યાં clause, to remove non-projectivity.

જ્યાં આજે ગાંધીસંગ્રહના આધુનિક સાદી ઢબના મકાન છે ત્યાં એ વખતે ખેતર હતું .- Where today is the modern simple style of Gandhi Collection, there was a farm.

5.4.2 Extraposed relative clause constructions:

If the relative clause modifying a noun phrase (NP) occurs after the verb group (VP), it leads to non-projectivity.



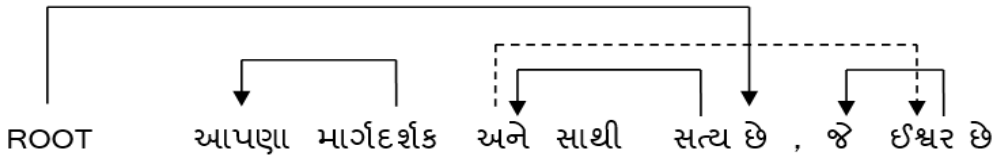
The girl who has an IQ 200, is studying with me

FIGURE 5.2(a): Extraposed relative clause construction

In the sentence in Fig.5.2 (a) non-projectivity occurs because તેમારી સાથે અભ્યાસ કરે છે, the relative clause modifying the NP એ છોકરી, is extraposed after the VP IQ ૨૦૦છે.

This class of constructions is not rigid as the extraposed relative clause can be moved next to the noun phrase, making it projective. However, the resulting projective construction is less natural than the original non-projective one. The reordered projective construction for the example sentence is એ છોકરી, તેમારી સાથે અભ્યાસ કરેછે, જેનો IQ ૨૦૦છે. - The girl who studies with me who has an IQ-200.This class of non-projective constructions accounts for approximately half of the total non-projective sentences.

Another Example,



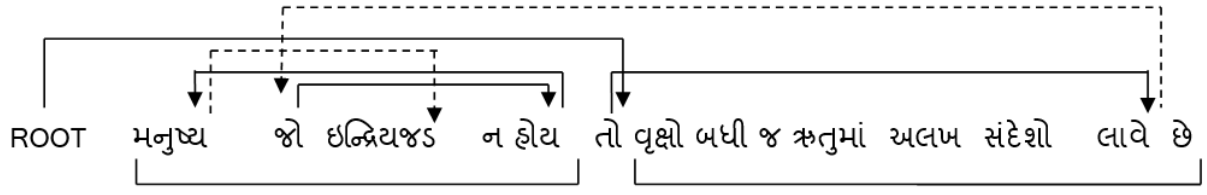
Truth, which is God, is our guide and companion

FIGURE 5.2(b): Extraposed relative clause constructions

In the sentence in Fig.5.2 (b) non-projectivity occurs because જે ઈશ્વર છે, the relative clause modifying the NP આપણા માર્ગદર્શક અને સાથી is extraposed after the VP સત્ય છે. The reordered projective construction for the example sentence is આપણા માર્ગદર્શક અને સાથી, જે ઈશ્વર છે, સત્ય છે. - Our guide and companion which is God is truth.

5.4.3 Intra-clausal non-projectivity:

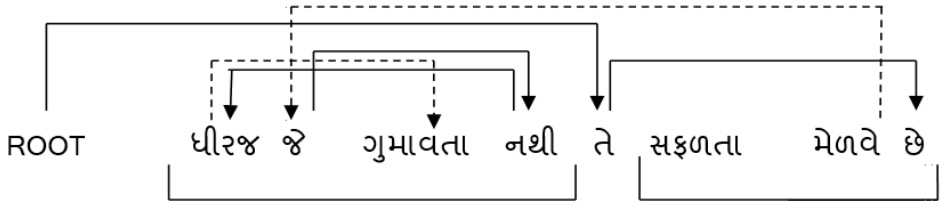
In this case, the modifier of the NP is a non-relative clause and is different from the above class 5.4.2



If humans do not have senses, the trees send a different message throughout the season.

FIGURE 5.3 (a): Construction with non-projectivity within a clause

In the example in Fig.5.3 (a) the NP મનુષ્ય and the phrase modifying it ઇન્દ્રિયજડ are separated by જો, a modifier of તો clause. Intra-clausal non-projectivity here is within the clause મનુષ્ય જો ઇન્દ્રિય જડ ન હોય. To remove non-projectivity, reordering of such sentences is possible by moving the non-modifier, so that it no more separates them. Here, moving જો to the left of મનુષ્ય takes care of non-projectivity thus making this class not rigid. The reordered projective construction is more natural. જો મનુષ્ય ઇન્દ્રિય જડ ન હોય તો વૃક્ષો બધી જ ઋતુમાં અલખ સંદેશો લાવે છે.If humans are not obsessed, the trees convey a different message throughout the season. Another Example,



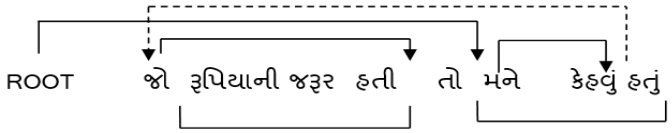
Patience that does not disappear gains success

FIGURE 5.3 (b): Construction with non-projectivity within a clause

In the example in Fig.5.3 (b), the NP ધીરજ and the phrase modifying it જે ગુમાવતા are separated by જે, a modifier of તે clause. Intra-clausal non-projectivity here is within the clause ધીરજ જે ગુમાવતા નથી. Here, moving જે to the left of ધીરજ takes care of non-projectivity thus making this class not rigid. The reordered projective construction is more natural. જે ધીરજ ગુમાવતા નથી તેસફળતા મેળવે છે -Those who do not lose patience gain success.

5.4.4 Paired connectives:

Paired connectives (such as જે- તે if –then) give rise to non-projectivity in the annotation scheme used.

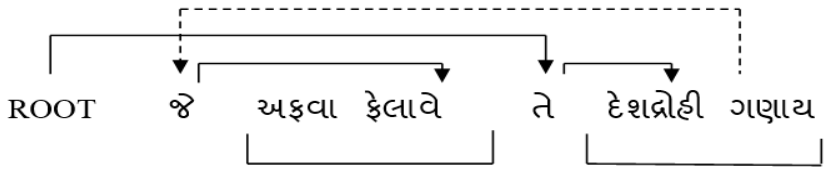


If [you] needed rupees then [you] should have told me

FIGURE 5.4(a): Paired connectives construction

As shown in Fig.5. 4 (a), the તો clause is modified by the જો clause in such constructions. Most of these sentences can be reordered while still retaining the meaning of the sentence: the phrase that comes after તો, followed by જો clause, and then તો. Here mentioning તોis optional. This sentence can be reordered and is not rigid. However, the resulting projective construction is not a natural one. મને કેહવું હતું જો રૂપિયાની જરૂર હતી [તો]-(you) should have told me if (you) needed rupees Connectives like જો can also give rise તો intra-clausal non-projectivity apart from inter clausal non-projectivity as discussed. This happens when the connective moves away from the beginning of the sentence.

Another Example



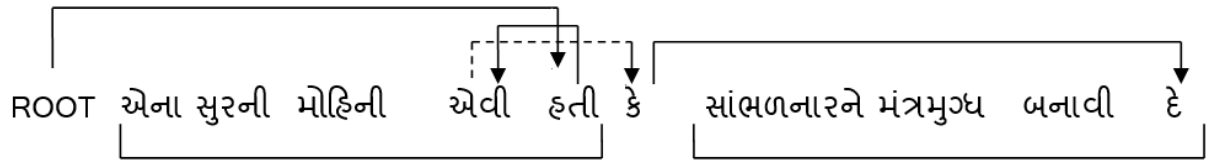
The rumour that spreads are called traitor

FIGURE 5. 4 (b): Paired connectives construction

This sentence can be reordered and is not rigid. However, the resulting projective construction is not a natural one. દેશદ્રોહી ગણાય જે અફવા ફેલાવે [તે] .-Called a traitor who spreads the rumour.

5.4.5 કે complement clause:

A phrase (including a VP in it) appears between the કે(that) clause and the word it modifies (such as આ(this), એવું (such), એવી રીતે (such), એટલું (this much), resulting in non-projectivity in the કે complement constructions. The verb in this verb group is generally copular. Since Gujarati is a verb final language, the complementizer clause (કે clause) occurs after the verb of the main clause, while its referent lies before the verb in the main clause. This leads to non-projectivity in such constructions.



Its charm was to make the listener fascinated

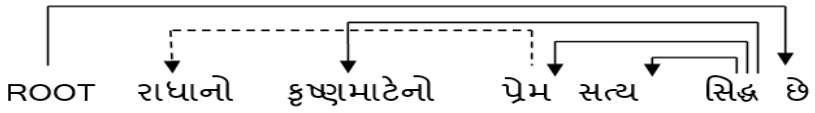
FIGURE 5.5: કે complement clause

In Fig.5.5, the phrase હતી separates એવી and the કે clause, resulting in non-projectivity.

These classes of constructions are rigid and non-projectivity can't be removed from such sentences. In cases where the VP has a transitive verb, the કે clause and its referent, both modify the verb, making the construction projective. For ex. In તેણે એ કહ્યું કે તે નહિ આવે., એ and the કે clause both modify the verb કહ્યું.

5.4.6 A genitive relation split by a verb modifier:

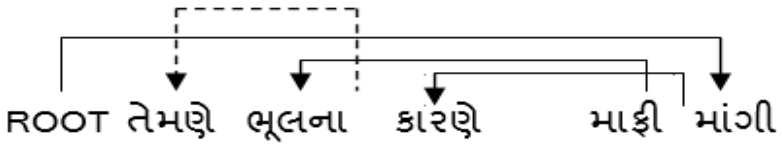
This is also a case of intra-clausal non-projectivity. In such constructions, the verb has its modifier embedded within the genitive construction.



Radha's love for Krishna is true

FIGURE 5.6(a): Genitive relation split by a verb modifier

In the example in Fig.5.6 the components of the genitive relation, રાધા and પ્રેમ are separated by the phrase કૃષ્ણ માટે. The sentence is not rigid and can be reordered to a projective construction by moving the phrase કૃષ્ણ માટે to the left of રાધા. It retains the meaning of the original construction and is also, a more natural one. કૃષ્ણ માટેનો રાધાનો પ્રેમ સત્ય સિદ્ધછે.- Radha's love for the Krishna is evident by itself. Another sentence is,

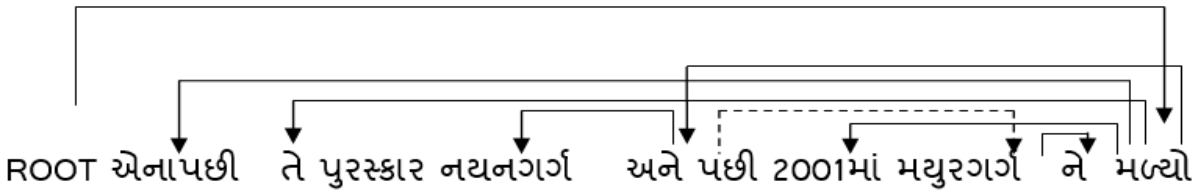


He apologized for the mistake

FIGURE 5.6(b) : Genitive relation split by a verb modifier

The sentence is not rigid and can be reordered to a projective construction by moving the phrase ભૂલ ના કારણે to the left of તેમણે. It retains the meaning of the original construction and is also, a more natural one. ભૂલના કારણે તેમણે માફી માંગી..- He apologized for the mistake

5.4.7 A phrase splitting a co-coordinating structure:

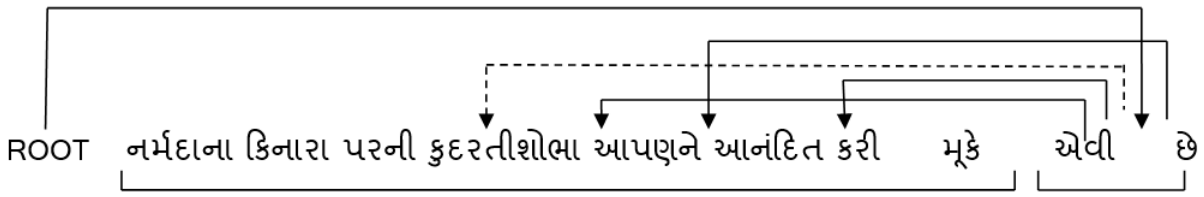


After this Nayan Garg [got it] and then, in 2001 Mayur Garg got it

FIGURE 5.7: A phrase splitting a co-ordinating structure

As seen in Fig.5.7, non-projectivity is caused in the sentence because, embedding of the Phrase 2001માંsplits the co-ordinating structure નયન ગર્ગ અને પછી મયુર ગર્ગ. These kinds of constructions can be reordered. So, they are not rigid. The projective constructions are more natural. એના પછી તે પુરસ્કાર નયન ગર્ગને અને પછી મયુર ગર્ગ ને૨૦૦૧માંમળ્યો.

5.4.8 Shared argument splits the non-finite clause:

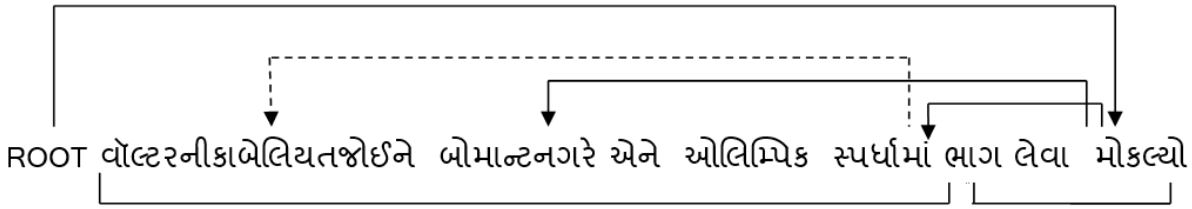


The beauty of the coast of Narmada is pleasing to us

FIGURE 5.8(a) : Shared argument splitting the non-finite clause

In the example in Fig.5.8 (a), આપણને is annotated as the argument of the main verb મૂકે એવી છે. It also is the shared argument of the non-finite verb આનંદિત કરી. It splits the non-finite clause નર્મદાના કિનારા પરની કુદરતી શોભા **આપણને** આનંદિત કરી. Through reordering, this sentence can easily be made into a projective construction, which is also the more natural construction for it. આપણને નર્મદાના કિનારા પરની કુદરતી શોભા આનંદિત કરી મૂકે એવી છે.- We have to rejoice on the shores of Narmada.

Another Example is,



Seeing Walter's ability, Bomant Nagar sent him to compete in the Olympics.

FIGURE 5.8(b): Shared arguments splitting the non-finite clause

Through reordering, this sentence can easily be made into a projective construction, which is also the more natural construction for it. બોમાન્ટ નગરે વૉલ્ટરની કાબેલિયત જોઈને એને ઓલિમ્પિક સ્પર્ધામાં ભાગ લેવા મોકલ્યો.- Bomant City saw Walter's ability and sent him to compete in the Olympics.

5.5 Conclusion

This study shows that non-projectivity in Gujarati is more or less confined to the classes discussed in it. There could be more types of non-projective structures in Gujarati which cannot have

occurred in the Treebank. The knowledge of such non-projective classes could possibly be used to enhance the performance of a parser. Further, linguistic insights into non-projectivity are often utilized in parsing to identify when to generate the non-projective arcs. The parser can have specialized machinery to handle non-projectivity only when linguistic cues belonging to those classes are active. The advantage of this is often that one needn't come up with formal complex parsing algorithms which give unrestricted non-projective structure.

CHAPTER - 6

Various Techniques of Dependency Paninian Parser

6.1 Introduction of Paninian Grammar and Parsing:

The goal of the Paninian approach is to construct a theory of human natural language communication that answers this question. Grammar is a part of theory of communication, is a system or rules that establish a relation between what the speaker decides to say and his utterance, and similarly what the hearer hears and therefore the meaning he extracts. The main problem the Paninian approach addresses is the way to extract karaka relation (syntactico-semantic relation) from a sentence. Gujarati is the official language of the union of India. It is a verb final language with free word order as SOV (subject, Object, verb) or OSV [23]. This can be seen in (a-f), where a) shows the constituents in the default order, and the remaining examples show some of the word order variants of a)

a) નિલય એ અલય ને પુસ્તક આપ્યું. “Nilay gave the book to Alay.” (S-IO DO-V)

b) નિલય એ પુસ્તક અલય ને આપ્યું.(S-DO- IO-V)

c) અલય ને નિલયે પુસ્તક આપ્યું.(IO-S-DO-V)

d) અલય ને પુસ્તક નિલયે આપ્યું.(IO-DO-S-V)

e) પુસ્તક નિલયે અલય ને આપી.(DO-S-IO-V)

f) પુસ્તક અલય ને નિલયે આપી.(DO-IO-S-V)

Guajarati is also a rich case of marking system. Although case marking is not obligatory. For example, in (a), while the subject and indirect object are marked explicitly for the ergative and the direct object is unmarked for the accusative.

Parsing is one among the most important tasks which helps in understanding the linguistic communication. It is useful in several natural language applications. Machine translation, anaphora resolution, word sense disambiguation, question answering, summarization is few of them. This led to the development of grammar driven, data-driven parsers. Most of the parsers for free word order languages are dependency based. The basic difference between a constituent-based representation and a dependency representation is the lack of non-terminal nodes in the latter. In Natural Language Processing, Parsing is a method of analysing the grammatical structure of sentences. Data driven and grammar driven approaches have been employed over the years for parsing of a natural language. The adjective data-driven means that progress in an activity is compelled by the data, rather than by intuition or by experience. The Grammar driven approach requires a deep knowledge of the formal grammar rules, syntax and semantics of the language. The Parsing is one of the principal steps involved in Machine translation. The machine translation of a natural language without parsing the sentences using syntax and semantics of can yield wrong results. For example, translate the sentences Gujarati using Google Translate shows the following output,

"Marry is Good in nature". (મેરી વર્તન માં સારી છે)

The result: "લગ્ન પ્રકૃતિ સારી છે." (Google translate)

similarly

"The Soldiers were searching for people with Helicopters."

The result: "સૈનિકો હેલિકોપ્ટરવાળા લોકો માટે શોધક હતા".

Parsing morphologically rich, free word order languages is a challenging task. Indian Languages are morphologically rich, free word order languages. As free word order languages can be handled better using the dependency-based framework, dependency annotation using Paninian framework is started for Indian Languages.

6.2 Paninian Grammatical model:

Indian language including Gujarati is morphological rich and flexible word order. For such languages syntactic subject-object are not able to explain the various linguistic phenomena. The Paninian grammatical model is primary concerned with: (1) how the information is coded and (2) how it can be extracted.

Two level of representation can be understanding in Gujarti language, one the actual sentences, two, what the speaker has in his mind. Paninian frame work has to important level: Karaka level and Vibhakti level [24].

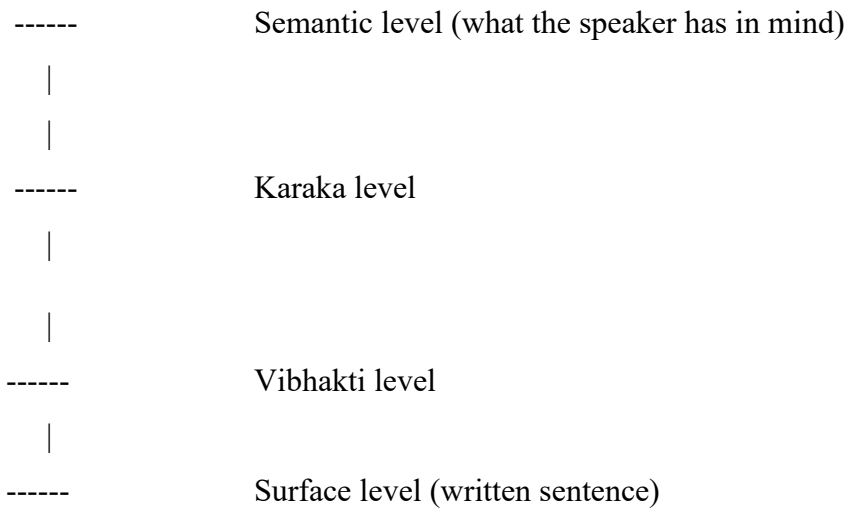


FIGURE 6.1: Levels of representation of the Paninian model

The surface level is uttered or written sentences. The Vibhakti level is the level at which preposition or postposition markers. The vibhakti level abstracts away from many minor differences among languages. There are only around six different types of karaka relation out of 28 relations, namely:

- k1: Karta (the most independent participant in the action)
- k2: Karma (the most desired by the Karta)
- k3: Karna(instrument)
- k4: Sampradaan (beneficiary)
- k5: Apaadaan (source)
- k7: Adhikaran (location)

6.3 Paninian Theory:

Paninian grammar is especially suited to free ordering languages. It makes use of vibhakti information for mapping to linguistic relation, and uses position information only secondarily. The Paninian frame work was originally designed quite two millennia ago for writing a grammar of Sanskrit, it has been adapted by us to affect modern Indian Language [23].

6.3.1 Karaka Relation for Gujarati Languages:

There is no simple mapping from vibhakti to linguistic relation between noun groups and verbs. [22,29]. The key to arriving at a solution is to identity appropriate relation is named Karaka relation.

The most important karaka-vibhakti mapping is depends on the verbs and its tense aspect modality (TAM) label. The mapping is represented by two structures: default karaka chart and karaka chart transformation.

The default karaka chart for a verb or a category of verb gives the mapping for the TAM label referred to as basic. It specifies the vibhakti permitted for the applicable karaka relation for the noun etc. when the verb has the basic TAM label.

B.1 કનક મોહન ને મારે છે. (Kanak Beats Mohan)

B.2 મોહન ને કનક મારે છે. (Kanak Beats Mohan)

B.3 મોહન કનક ને મારે છે. (Mohan Beats Kanak)

B.4 કનક ને મોહન મારે છે. (Mohan Beats Kanak)

The default karaka chart for 3 of the karakas is given in Table 6.1. This explains the vibhaktis in sentences B.1 to B.4. As kanak is agent in B.1 and B.2, and Mohan B.3 and B.4, and agent is that the most independent for the action (beat), it's expressed by means of the karta karaka and the remaining nominal by karma karaka. For the karta karaka, ∅ vibhakti is used with રે છે –TAM label in B.1 to B.4 as explained in Table 6 .1.

TABLE 6.1: Default karaka chart

Karaka	**Vibhakti**	**Presence**
Karta	∅	Mandatory
Karma	∅ *or* ને	Mandatory
Karana	થી or દ્વારા	optional

C.1 માધવે ફળ ખાધું (Madhav ate the fruit)

C.2 માધવ ને ફળ ખાવું પડયું (Madhav had to eat fruit)

C.3 માધવ થી ફળ ના ખવાયું (Madhav couldn't eat fruit)

Table 6.2 gives some transformation rules for the default mapping for Gujarati. It explains the vibhakti within the above sentences. (assuming Madhav is that the karta and fruit is that the karma). As Explained by Table 6.2, Karta takes 'એ 'vibhakti in C.1 because of TAM label 'ધુ' (in the main verb group ખાધું,) 'ને 'in C.2 because of TAM label 'ઉ 'ચુ–(in ખાવું પડયું) and 'થી'in C.3beacause of TAM label 'આ'ચુ-(in ખવાયું)

TABLE 6.2: Transformation rules

TAM label	**Transformed Vibhakti for karta**
ધુ	એ
ઉ -ચુ	ને
આ-ચુ	થી or દ્વારા (karta is optional)

The Paninian theory outline above. (i.e., karaka charts and karaka charts transformation) are used to generate the sentences given above as we've seen. For that constrains are given below:

1. Each mandatory karaka in the karaka chart for each verb group, is expressed exactly once.
2. Each optional karaka within the karaka chart for each verb group, is expressed at the most once.
3. Each source word group satisfies some karaka relation with some verb (or another relation).

6.3.2 Karaka to vibhakti Mapping for Gujarati:

If the sentences containing more than one verb group, then the theory comes from complex sentences. For that consider the following sentences:

M1: માણસે નાવ બનાવવા માટે લાકડી કાપી.

(The Man Cut wood to make a boat.)

M2: યોગી એ ચોકલેટ વહેંચીને ખાધી.

(yogi distributed the chocolate and ate it.)

M35: ફળ કાપવા માટે યોગી એ ચપ્પુ લીધું.

(To cut the fruit, Yogi took a knife.)

In M1 માણસ (The man) is Karta of Both the Verbs: બનાવવું (made) and કાપવું (Cut), However it occurs only once. The Problem is identified which verb control its vibhakti. In M2 karta યોગી (Yogi) and the karma ચોકલેટ (Chocolate), both are shared by the two verbs વહેંચવું (distribute) and ખાધી (eat). In M3 the karta યોગી (Yogi) is shared between the two verbs, and ચપ્પુ (knife) the karma karaka of લીધું (take)is the karana (instrumental) karaka of કાપ્યું (cut). The observation of the matrix or main verb rather than the intermediate verb controls the vibhakti of the shared nominal is true in the above sentences. A different vibhakti can be used for the same semantic relation with a given verb in a different sentence. A different vibhakti is used each time (ø , એ , ને,

થી which represent nominative, ergative, accusative, ablative respectively). The intermediate verbs have their TAM labels just like other verbs. As usual, these TAM labels have transformation rules that operate and modify the default karaka chart. In particular, the suggested transformation rule for the two labels is given Table 6.3. The transformation rule with ને in Table 6.3 says that karta of the verb with TAM label ને must not be present in the sentence and the karma is optionally present.

TABLE 6.3: More transformation rules (for complex sentences)

TAM label	Transformation
ઈ- ને	karta must not be present. Karma is optional
વું	Karta and karma are optional
તા	Karta and karma are optional

6.4 Verb frame for Gujarati language:

Verbs are the most important grammatical category in a language. Actions, activities and states are denoted with the assistance of the verbs. The arguments of the verb specify different participants required by the verb. Verbs play a significant role in interpreting the sentence meaning therefore, the study of verb argument structure and their syntactic behaviour will provide the required knowledge domain for intelligent Natural Language Processing applications. A demand frame or karaka frame for a verb indicates the demands the verb makes. It depends on the verb and its tense, aspect and modality (TAM) label. A mapping is specified between karaka relations and vibhaktis (post-positions, suffix). The notion of karaka relations is central to the Paninian framework [69,70]. The karaka relations are syntactico-semantic relations between the verb and the other constituents of the sentence. They capture a certain level of semantics. The approach uses case markers (vibhakti information) for mapping the relation between the verb and its arguments.

The verb frames developed following this framework show the mandatory *karaka* relations for a verb. Each verb can have multiple senses and for each sense of a verb there can be a number of possible frames. Verb Frame is represented in a tabular form.

A verb frame shows:

(1) Karaka (dependency) relations;
(2) Necessity of the argument, i.e., whether an argument is mandatory (m) or desirable (d);
(3) Vibhakti (Gujarati postpositions/case markers taken by the arguments);
(4) Lexical category of the arguments, i.e., noun, verb and adjective, etc., represented by n, v, and adj, respectively.

Example 1:

રામ સીતા ને માટે સંદેશ મોકલે છે . (Ram sends the message for the Sita)

TABLE 6.4: Verb frame1: મોકલે છે (sends)

Arc-labeled	**Necessity**	**Vibhakti**	**Lexical -type**
k_1	m	0	n
k_2	m	0	n
k_4	m	ને માટે	n

Example 2:

માતા-પિતા બાળકોને શાળા એ મોકલે છે. (Parents sends their children to school)

TABLE 6.5: Verb frame2: મોકલે છે (sends)

Arc-labeled	**Necessity**	**Vibhakti**	**Lexical -type**
k_1	m	0	n
k_{2p}	m	એ	n
k_2	m	ને	n

In the above Gujarati example sentence 1 and 2, the verb મોકલે has the same sense. i.e., in example 1 the verb મોકલે 'send' is taking the following arguments: karta, Sampradaan and karma. In example 2 the verb મોકલે 'send' is taking the following arguments: karta, karma and goal. Here it can be notice that there is a difference in the set of dependency relations of the argument taken by verb મોકલે 'send' in the example 1and 2. This shows that the same sense of verb can take multiple frames. There exist a finner distinction in the sense of મોકલે in example 1 and 2. i.e.; in example1, it is an object (સંદેશ – 'message') that is being sent, so the verb મોકલે becomes a ditransitive verb here. (A *Ditransitive* Verb is one that takes both a direct object and an indirect object.) Whereas in example 2, it is an individual (બાળકો- 'Children') who is being sent so the verb મોકલે becomes a causative verb here. (The causative verb meaning is**,** when we do not carry out an action ourselves, but are responsible for the action being performed.)

6.5 Dependency Parsing:

Dependency graphs represent words and their relationship to syntactic modifiers using directed edges [17, 28, 71, 97]. Dependency parsing can be broadly divided into grammar-driven and data-driven dependency parsing. Most of the modern grammar-driven dependency parsers parse by eliminating the parses which do not satisfy some set of grammatical constraints. Fig.6.2 shows a dependency graph for the sentence, "John hit the ball with the bat"

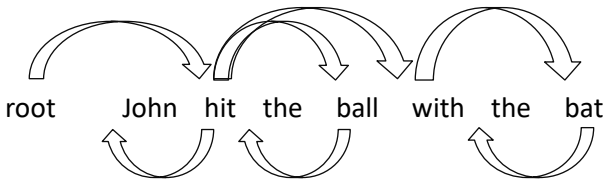


FIGURE 6.2: Projective Dependency Tree (English)

This example belongs to the special class of dependency graphs that only contain projective (also known as nested or non-crossing) edges., a projective graph is one that can be written with all words in a predefined linear order and all edges drawn on the plane above the sentence, with no edge crossing another [28]. Consider the sentence, “John saw a dog yesterday which was a Yorkshire Terrier”. Here the relative clause “which was a Yorkshire Terrier” and the noun it modifies (the dog) are separated by a temporal modifier of the main verb. There is a no way to draw the dependency graph for this sentence in the plane with no crossing ages, as in Fig.3.6 in languages with flexible word, such as Indian language like Gujarati, Hindi, Telugu etc. non-projective dependency are more frequent.

FIGURE 6.3: Non-Projective Dependency Tree (English)

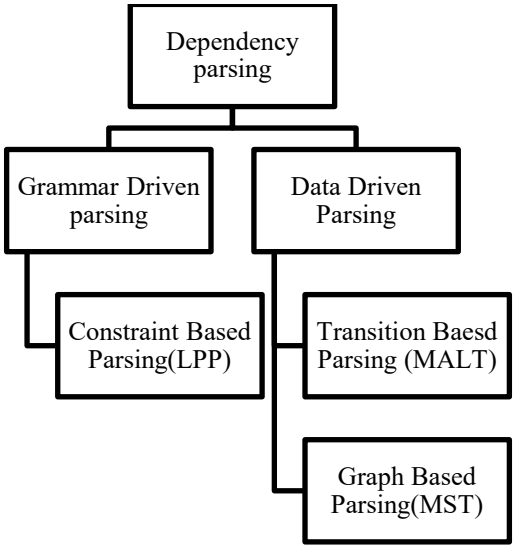


FIGURE 6.4: Dependency Chart

6.6 Approaches to Dependency Parsing:

Bharat Ram Ambati and his team used to two data drive parsers Malt, and MST for his experiments. Malt may be a classifier-based Shift/Reduce parser. It uses arc-eager, arc-standard, Covington projective and Covington non-projective algorithms for parsing. History-based feature models are used for predicting subsequent parser action [18]. Support vector machines are used for mapping histories to parser actions. It uses graph transformation to handle non-projective trees. Maximum Spanning Tree uses Chu-Liu-Edmonds for non-projective parsing and Eisner's algorithm for projective parsing. It uses online large margin learning because the learning algorithm.

Data driven parsers need large amount of manually annotated parsed data which is named a Treebank. The provision of such data for Gujarati is limited. On the other hand, most of the modern grammar-driven dependency parsers parse by eliminating the parses which don't satisfy the given set of constraints. They require rules to be developed for every layer. In the work of Akshar Bharati et al [13], the grammar driven approach is complemented by a controlled statistical strategy to achieve high performance and robustness.

Developing a grammar driven parser requires a deep knowledge of the dependency relations of the phrases. The creation of rules to disambiguate the relations is extremely challenging task.

6.7 Constraints Parser:

A parse may be a sub graph of constraints graph containing all the nodes the constraints graph and satisfying following condition

- C1: For each of the mandatory Karakas in a Karaka chart for each demand group (verb group) there should be exactly one out going edge labeled by the karaka from the demand group
- C2: For each the optional karakas in a karaka chart for each demand group, there should be at the most one out going edge labeled by the karaka from the demand group.
- C3: There should be exactly one incoming arc in to each source group.
- If several sub graph of constraints graphs satisfy the above condition it means there are multiple parses and therefore the sentences are ambiguous. If not, a parse is named solution graph [20]**.**

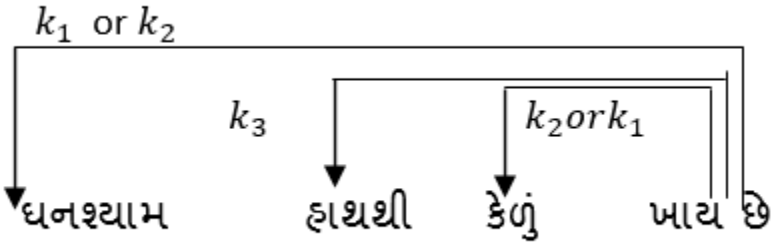


FIGURE 6.5: Constraint graph for the given sentences

6.8 Constraint Based Parsing Using LPP:

Parsing can be obtained from the constraint graph using integer programming [23]. A constraint graph is converted into an integer programming problem by introducing a variable x for an arc from node $i \ to \ j$ labelled by karaka k in the constraint graph such that there is a variable. The variable takes their values as 0 or 1. Equality and inequality constraint in the integer programming problem can be obtain from the condition as below:

1. For Each demand group (verb) i for each mandatory karaka k the following equality must hold:

$$O_{i,k} \ = \ \sum_j x_{i,k,j} = 1$$

Note that $O_{i,k}$ stands for the equation from given demand word i and karaka k ,thus there will be as many equations as combination of i and k .

2. For each demand group i, for each of its optional karaka k the following equality must hold

$$P_{i,k} : \sum_j x_{i,k,j} \leq 1$$

3. For each of the source group (noun) j, the following equality must hold

$$T_j : \sum_{i,k} x_{i,k,j} = 1$$

Thus, their will as many equations as there are source word.

6.8.1 Implementation:

Projective:	ઘનશ્યામ	હાથથી	કેળું	ખાય છે.
	k1	k3	k2	M.V.
	a	b	c	A

Eng.: Ghanshyam eats bananas by hand.

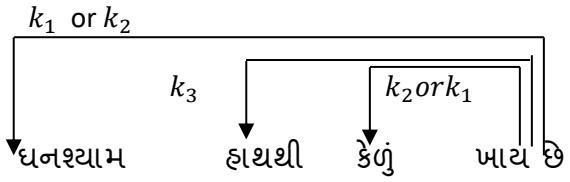


FIGURE 6.6: Constraint Graph

Constraint C1:

$O_{A,K1} : x_{A,K1,a} + x_{A,K1,c} = 1$ $O_{A,K2} : x_{A,K2,a} + x_{A,K2,c} = 1$

Constraint C2:

$P_{A,K3} : x_{A,K3,b} \leq 1$

Constraint C3:

$T_a : x_{A,K1,a} + x_{A,K2,a} = 1$

$T_b : x_{A,K3,b} = 1$

$T_c : x_{A,K1,c} + x_{A,K2,c} = 1$

Take $x_{A,K1,a} = y_1$, $x_{A,K1,c} = y_2$, $x_{A,K2,a} = y_3$, $x_{A,K2,c} = y_4$, $x_{A,K3,b} = y_5$

We get

$O_{A,K1} : y_1 + y_2 = 1$

$O_{A,K2} : y_3 + y_4 = 1$

$P_{A,K3} : y_5 \leq 1$

$T_a : y_1 + y_3 = 1, T_b : y_5 = 1 \ , \ T_c : y_2 + y_4 = 1$

The cost function to be minimized is:

Min Z = $y_1 + y_2 + y_3 + y_4 + y_5$

Subject to, $y_1 + y_2 = 1$

$y_3 + y_4 = 1$

$y_5 \leq 1$

$y_1 + y_3 = 1$

$y_5 = 1$

$y_2 + y_4 = 1$

Result:$y_1 = 1$,$y_4 = 1$,$y_5 = 1$

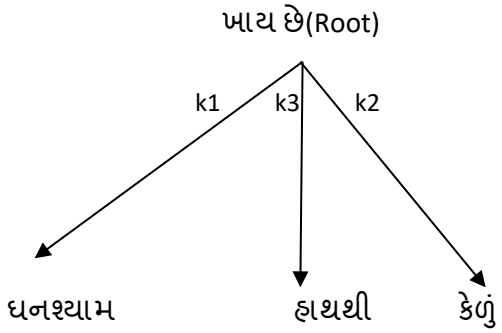


FIGURE 6.7: Parsing for the Projective Sentences

Non-Projective: રામે ફળ ખાઈને મોહનને રમકડું આપ્યું.

k1	k2	V. Modi	k4	k2	M.V.
a	b	B	c	d	A

Eng.: Ram ate the fruit and gave the toy to Mohan.

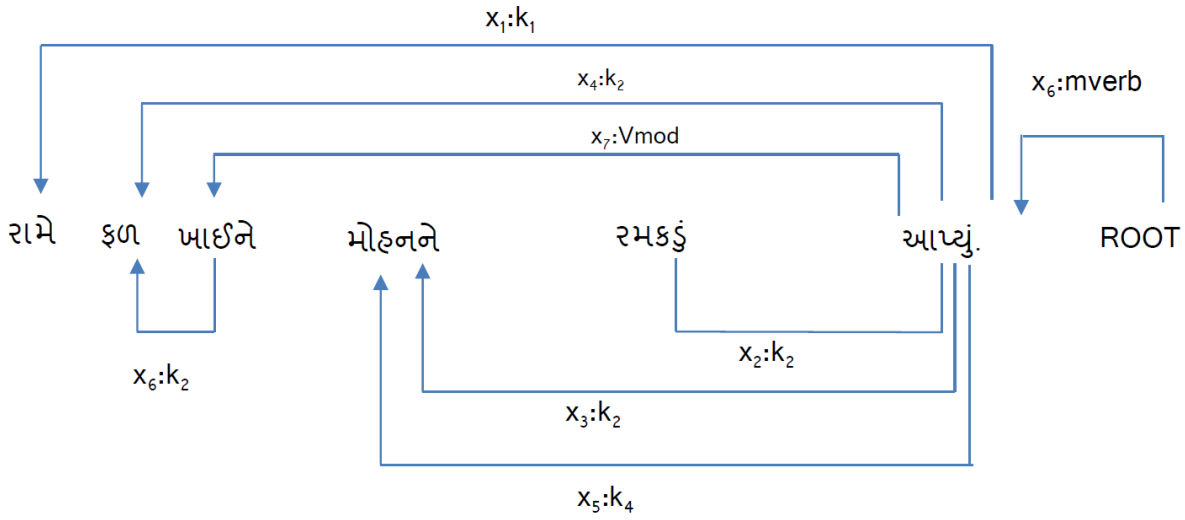


FIGURE 6.8: Constraint graph for the Sentences

Constraint C1: (for the verb "give'')

$O_{A,K1} : x_{A,K1,a} = 1$

$O_{A,K2} : x_{A,K2,b} + x_{A,K2,c} + + x_{A,K2,d} = 1$

$O_{B,K2} : x_{B,K2,b} = 1$ (for the verb "eat'')

Constraint C2:

$P_{A,K4} : x_{A,K4,c} \leq 1$

Constraint C3: (incoming arc into source C3)

$T_a : x_{A,K1,a} = 1$

$T_b : x_{A,K2,b} + x_{B,K2,b} = 1$

$T_c : x_{A,K2,c} + x_{A,K4,c} = 1$

$T_d : x_{A,K2,d} = 1$

Take, $x_{A,K1,a} = y_1$, $x_{A,K1,c} = y_2$,$x_{A,K1,d} = y_3$, $x_{A,K2,a} = y_4$,

$x_{A,K2,c} = y_5$ $x_{A,K2,d} = y_6$ $x_{A,K4,c} = y_7$, $x_{B,K2,b} = y_8$,

$x_{A,K2,b} = y_9$

We get

$O_{A,K1} : y_1 + y_2 + y_3 = 1$

$O_{A,K2} : y_4 + y_5 + y_6 = 1$

$O_{B,K2} : y_8 = 1$

$P_{A,K4} : y_7 \leq 1$

$T_a : y_1 = 1,$

$T_b : y_8 + y_9 = 1$,

$T_c : y_5 + y_7 = 1$

$T_d : y_6 = 1$

The cost function to be minimized is:

Min Z = $y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7 + y_8 + y_9$

Subject to,

$y_1 + y_2 + y_3 = 1$

$y_4 + y_5 + y_6 = 1$

$y_8 = 1$

$y_7 \leq 1$

$y_1 = 1$

$y_8 + y_9 = 1$

$y_5 + y_7 = 1$

$y_6 = 1$

Result: $y_1 = 1$,$y_6 = 1$, $y_7 = 1$,$y_8 = 1$

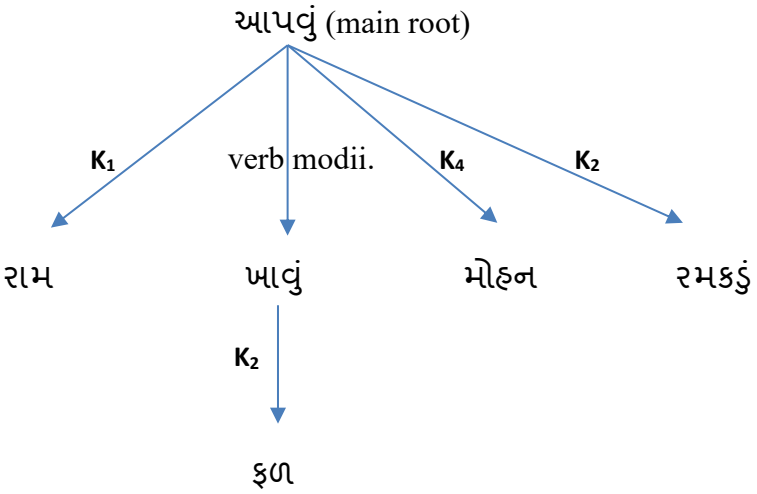


FIGURE 6.9: Parsing for the Non-Projective Sentences

TABLE 6.6: Verb Frame આપ્યું (give)

arc-label	necessity	vibhakti	lextype	arc-pos
K1	m	એ	n	l
K2	m	ને	n	l
K3	d	થી	n	l
K4	d	ને	n	l

TABLE 6.7: Verb frame ખાવું (eat)

arc-label	necessity	vibhakti	lextype	arc-pos
K1	m	0	n	l
K2	m	0	n	l

6.9 Constraint Parser Using Matching Graph:

A constraints graph which does not have any optional karaka, it can reduce the constraints graph to a Bipartite Graph. Finding a solution graph to the Constraint graph reduces to finding a maximal matching in the Bipartite Graph.

In the Mathematical field of graph theory, a bipartite graph is a graph whose vertices can be divided into two disjoint and independent sets and such that every edge connects a vertex into one in.

Definition 1: Bipartite Graph

A Bipartite Graph $G = (U, V, E)$ is defined as,
U : Set of nodes $U = u_1, u_2, u_2 \dots \dots \dots u_n$
V : Set of Node $V = v_1, v_2, v_2 \dots \dots \dots v_n$
E : edges between U and V and $U \cap V = \emptyset$

Definition-2: Matching

Given a graph G = (V, E), M is a matching in G if it is a subset of E such that no two adjacent edges share a vertex.

Definition -3: Maximal Matching

A maximal matching may be a matching to which no more edges will be added without increasing the degree of one of the nodes to two. It is a local maximum.

Definition-4: Maximum Matching

A maximum matching is a matching with the largest possible number of edges; it is globally optimal.

6.9.1 Constructing Initial Bipartite Graph:

To finding the solution graph to finding a matching, we first construct the bipartite graph. The Bipartite graph is constructed in three stages:

1. For every source node s in the constraint graph, form a node s in U.
2. For every demand node d in the constraint graph and every mandatory karaka kin the karaka chart for d, form a node v in V.
3. For every edge labeled by karaka k in the constraint graph, create an edge between nodes in V to s in U.

For example, for the constraints graph in the Fig. 6.5 (assuming that the optional karana is mandatory) the bipartite graph in Fig. 6.10

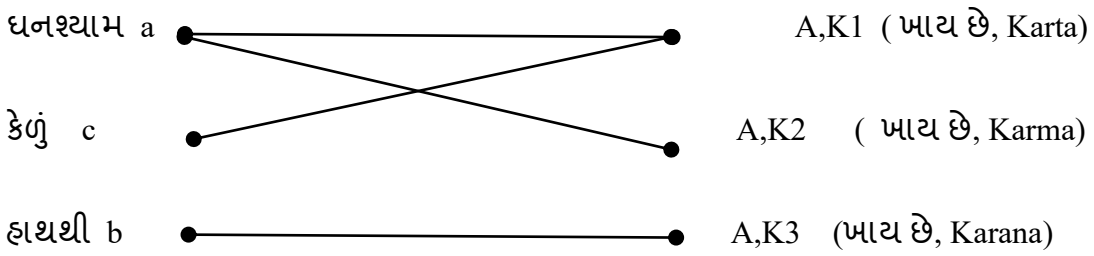


FIGURE 6.10: Bipartite graph for constraint graph in Fig 6.5

6.9.2 Parsing as Bipartite Graph Matching Problem:

The parsing problem now can be reduced to a bipartite graph matching problem. The bipartite graph (BG) G is defined as a three tuple (D, S, E) where D= {(di, k_j), for each demand group di and its Karaka demands k_j's in its demand frame}, S is the set of all source groups {s}, E is the set of edges from D to S, i.e., E= {(d_i, kj, s_t)}, if an attachment is possible from (di, kj) to a source group s_t and D∩S=ϕ. For a weighted BG, E is redefined as E= {(d_i, k_j, s_t, w)}, where w is called the weight of the edge. We have set a weight of 2 for mandatory Karakas and 1 for desirables, so that a mandatory Karaka surely gets an arc [12]. Now we define a matching M on the bipartite graph as M⊑E with the property that no two edges of M have a common node (one-to-one mapping). The matching problem is to find a maximal matching of G, i.e., matching with the

largest number of edges. A maximal matching is called complete if every node in D and S has an edge (one-to-one and onto).

6.9.3 Matching Graph:

- A matching of a bipartite graph $G = (U, V, E)$is a subset of edges with the property that no edges of M share the same node. The matching problem is to find a maximal matching of, that is, a matching with the largest number of edges.
- A maximal matching is called complete matching if every node in U and V has an edge. There are two maximal complete matchings of the graph in Fig.6.10. There are shown in Fig.6.11.

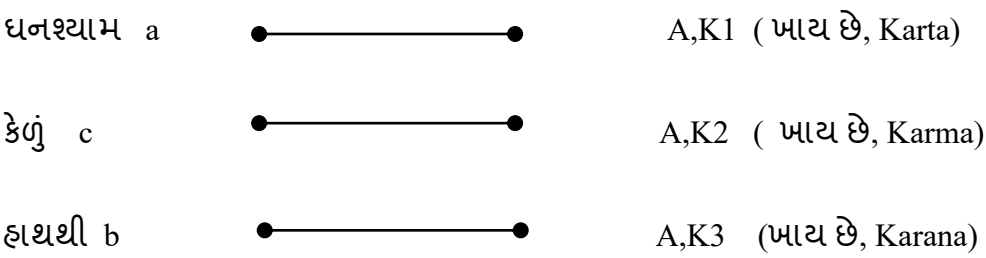


FIGURE 6.11 (a): possibility of Parse structure

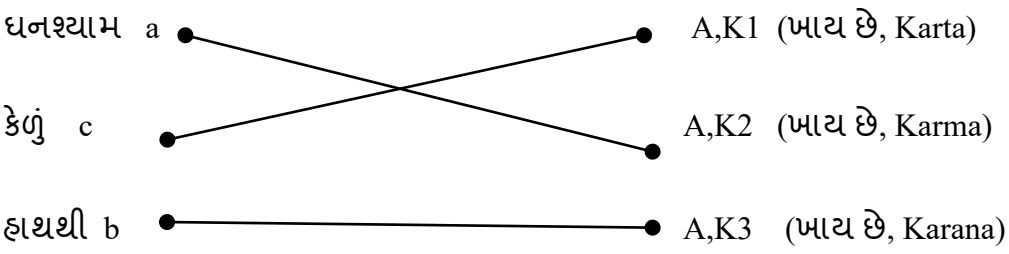


FIGURE: 6.11 (b) another possibility of Parse structures

FIGURE: 6.11: Maximal (Complete) Matching of Bipartite Graph of Fig 6.10

Now show that finding a maximal matching in a bipartite graph is the same as finding a parse in a constraint graph. Let M be the maximal matching of a bipartite graph G. If M is complete, it represents a Parse. For an edge between a node (d, k) in V and node S in U, it represents a demand for karaka k of the demand node d being satisfied by the source node s. Since all the nodes in V have exactly one edge in M, the constraints C1 is satisfied. All the nodes in U have exactly one edge in M, constraints C3 is satisfied. There is no optional karaka, C2 is satisfied, its trivially. If

M is not complete that is, it does not have an edge on at least one node in U or V then G does not have a parse. If node V in does not have an edge, it is a violation of constraints C1, otherwise violation of constrains C3 therefore M is not a parse. Consequently, some node in U or V would not have an edge thus any other maximal matching would also not give us parse.

6.10 Conclusion:

In this work to show, how computational Paninian Gujarati grammar (for projective and non-projective sentence) can be used for parsing word order languages. This is based on translating Gujarati grammatical constraint to LPP constraints. A solution of these constraints produces a parse. The result is powerful and versatile parser in Gujarati language.

For the LPP it can be solved by LINGO for the projective and non-projective sentence it gives correct answer but for the non-projective sometimes it gives not correct answer. For the maximal matching we can use only where optional karaka is absent so for the projective sentence maximum bipartite method is better also for the non-projective as discuss in chapter 5 is convert into projective sentence and using maximum matching of bipartite graph.

CHAPTER-7

Conceptual Graph Based Parsing

7.1 Overview

7.1.1 Knowledge Representation Issues in AI:

Artificial Intelligence as technology has always fascinated human beings. There have been multiple science fiction novels and movies where AI-powered systems such as Robots can think, act, understand complex information, and make smart decisions based on it. However, one concept that one must understand before creating that level of Artificial Intelligence is rather psychological or biological. Knowledge Representation in Artificial Intelligence refers to that concept where ways are identified to provide machines with the knowledge that humans possess so that AI systems can become better. As it is a universal fact that more a person knows a subject matter, the chances of taking a correct action or decision will be higher. This gives the AI developers who are in the quest of making the AI systems smarter a task at hand- to represent the knowledge of the human world in a way that machines can understand and can make the AI systems smarter to solve complex real-world problems. The problem is that we humans process information in a highly complex manner.

We have concepts that are completely alien to the machine, such as intuition, intentions, prejudices, beliefs, judgments, common sense, etc., while some knowledge is straight forward such as knowing certain facts, general knowledge regarding objects, events, people, academic disciplines, language among other straight-forward things that machines have been able to comprehend with a level of success. With Knowledge Representation and Reasoning (KR, KRR), now we have to represent this information in a machine-understandable format and make the AI

system truly intelligent. Here knowledge will mean providing and storing the information regarding the environment, reasoning will be deducing this stored information, and intelligence will mean taking decisions and actions based on knowledge and reasoning.

7.1.2 Conceptual Graphs:

- A conceptual graph is a finite, connected bipartite graphs. The nodes of the graphs are concepts (represented as a box) or conceptual relations (represented as an ellipse) [45]

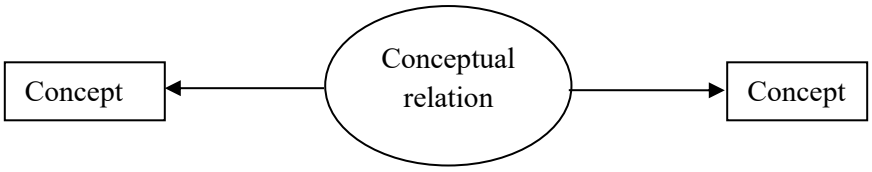


FIGURE 7.1: Representation of CG

- Conceptual graphs do not have labelled arcs. Instead, conceptual relations nodes represent relation between concepts. Concepts can only have arcs to conceptual relations and vice-versa. This simplifies the representation of knowledge. In a conceptual graph, a relation of arity *n* is represented by a conceptual relation node having *n* arcs and different arities. Each conceptual graph represents a single proposition.

Proposition 1: A bird flies

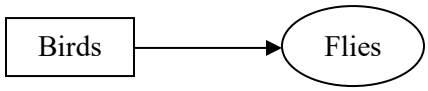


FIGURE 7.2: Flies is a 1-ary relation

Proposition 2: An apple taste sweet

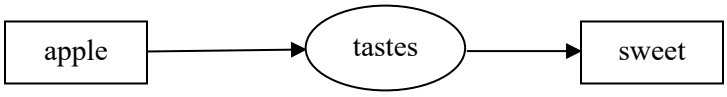


FIGURE 7.3: Taste is a 2-ary relation

Proposition 3: Apurva likes computer and AI

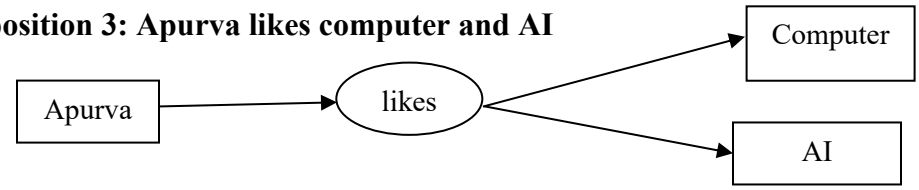


FIGURE 7.4: Likes is a 3-ary relation

Proposition 4: Mary gave John the book

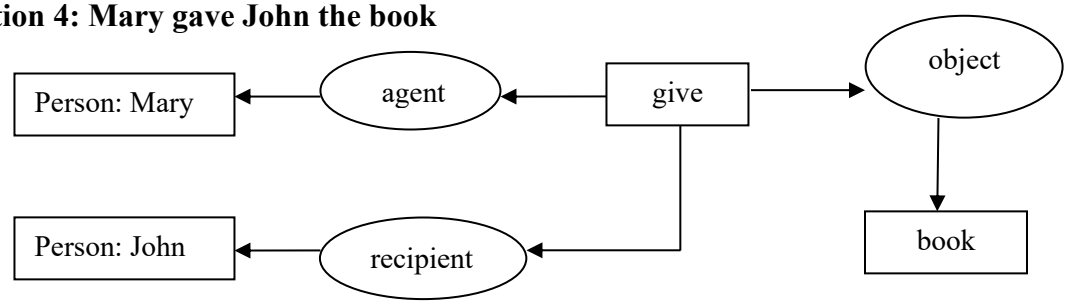


FIGURE 7.5: CG for the Given Sentences

Conceptual Graphs are **finite**, **connected**, **bipartite** graphs.

- **Finite**: because any graph can only have a finite number of concepts and conceptual relations.
- **Connected:** because two parts that aren't connected would simply be called two conceptual graphs.
- **Bipartite**: because there are two different kinds of nodes: **concepts** and **conceptual relations**, and every **arc** links a node of one kind to a node of another kind

7.1.3 Perception:

- Perception is that the process of building a working model that represents and interprets sensory input'. The reception of sensory input, 'a mosaic of precepts', is converted into concepts. Concrete concepts – that have associated precepts - Abstract concepts – that do not have any associated precepts.

- **Example** Consider the sentence:

Mary gave John (1) the boring book (2) authored by Tom & Jerry (3)

There are three main parts: (1), (2), and (3)

(1) Mary gave John

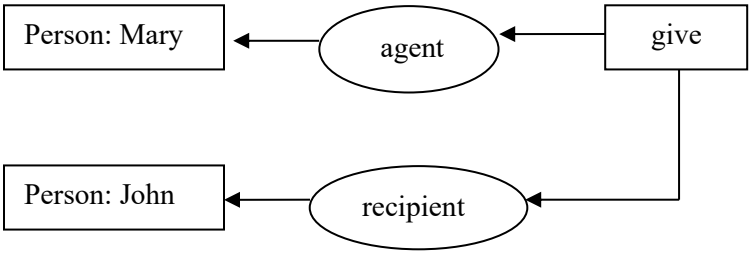


FIGURE 7.6 (a): CG Perception for the given sentence

Both relation nodes have two arcs each and are referred to as expressing a 2-ary or binary relation between the two concepts.

(2) the boring book

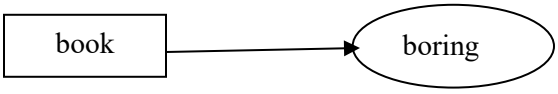


FIGURE 7.6 (b): CG Perception for the given sentence

The relation node has only one arc and thus refers to a 1-ary or unary relation

(3) authored by Tom & Jerry

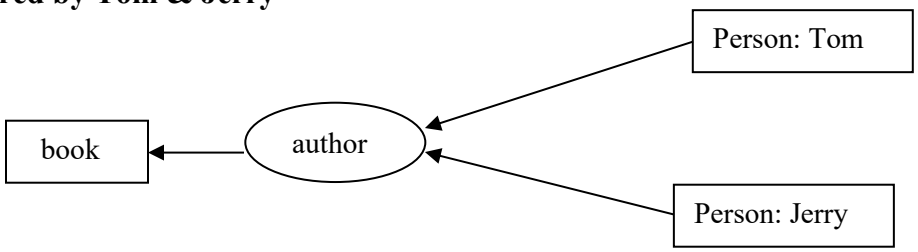


FIGURE 7.6 (c): CG Perception for the given sentence

The relation node has 3-arcs and is referred to as expressing 3-ary or ternary relation

7.1.4 Concept Nodes:

In CG theory, 'every concept is a unique individual of a particular type'. Concept nodes are labelled with descriptors or names like "dog", "cat", "gravity", etc. The labels refer to the class or type of individual represented by the node. Each concept node is used to refer to an individual concept or a generic concept. In CG theory we've a relation called: name.

7.1.5 Types in Conceptual Graphs:

In a conceptual graph, every concept is a unique individual of a specific type. The types are organized during a hierarchy.

- Each concept box is labeled with the names of the type and the individual. The type and the individual are separated by a colon.
- In a concept box the type label indicated the type (or class) of the individual concept represented by that concept node. If the concept node doesn't indicate the individual, then it means the concept node represents an unspecified concept of the given type.
- Boxes with the same type of label represent concepts of the same type. However, they'll or might not represent an equivalent individual concept**.**

7.2 Introduction:

The Introduction of Conceptual Graph (CG) by Sowa [89,91], the CGs are applied to several knowledge-based models comparison techniques like knowledge management task like information retrieval and text mining, database interface [93], generation of referring expression [31], implementing semantic interpreter [93], Comparison of personal ontology [36], ontology similarity measure [32] and so on. During this study, here propose a Conceptual Graph to the Parsing of the natural language Guajarati. Gujarati is spoken by more than 3 million. Gujarati Languages are included within the 23 official languages of India and are incorporated within the Indian constitution. Parsing of a natural language with respect to the syntactic and semantic knowledge of the language is important. Sowa [89], the original author of the conceptual graph theory who formed the idea for the Conceptual Structure, said: "A conceptual graph has no meaning in isolation. Only through the semantic network are its concepts and relations linked to context, language, emotion, and perception."

Conceptual Graphs are a visual, logic-based knowledge representation formalism. They encode ontological knowledge during a structure called support. The support contains concept type hierarchy and relation type hierarchy, a set of individual markers that refer specific concept and generic marker, denoted by * which refer to an unspecified concept in the application domain. A CG is structure that depicts factual information about the background contained in its Support. This knowledge is presented during a visual manner as an ordered Bipartite Graph, whose nodes are labelled with elements from the Support.

Definition: 7.2.1: Conceptual Graph

Conceptual Graph: A (simple) conceptual graph (CG) [30,33] is a triple SG: $[S, G, \lambda]$ where

- $S = (T_C, T_R, I, *)$ is a support.
- $G = (V_C, V_R, N_G)$ is an ordered bipartite graph.
- λ: $G \rightarrow S$ is a labelling function.

A relation node r is labelled by type (r) and element of T_R called its type.

The degree of r must be equal to the arity of type(r).

i.e. $\forall\, r \in V_R\ , \lambda(r) \in\ T_R^{dG(r)}, , \lambda(r) = t_r$

$\forall\, c \in V_C\ \ , \lambda(c) \in T_C * (I \cup \{*\})\ , , \lambda(c) = \big(type(c), marker(c)\big),$

$type(c) \in T_{C,}\ ,\ \ marker(c)\ is\ an\ element\ of\ M.$

$$marker(c) = \{^{*,generic}_{I,Individual}$$

Definition : 7.2.2: Ordered Bipartite Graph

A graph $G(V_C, V_R, N_G)$ is called an ordered bipartite graph [33] if V_C and V_R are finite disjoint set, where $V = V_C \cup V_R$ is the vertices set of G, and $N_G : V_R \rightarrow V_C^+$ is a mapping, where V_C^+ is the set of all finite non-empty sequence over V_C, for $r \in V_R$ with $N_G(r) = c_1, c_2 \ldots \ldots c_{k,} d_G(r) = k$ is the degree of r in G and $N_G^i(r) = c_i$ is the ith neighbor of r in G. $i.e; V = N_G^i(r) iff \{r, v\} \in E_G\ and\ l(\{r, v\}) = i$

The set of (distinct) neighbours of r is denoted by $\overline{N_G}(r)$. The set of edges of G is given by E_G: $\{(c, r)/c \in V_C\ , r \in V_R\ \ and \exists i$ such that $N_G^i(r) = c\}$.

Definition:7.2.3: Support

S: $(T_C, T_R, I, *)$ is a Support [33,90] where:

- T_C is a finite partially ordered set, $(T_C, \leq)$, of Concept type
- T_R is a Finite relation type partition into k partially ordered set $(T_R^i, \leq) i = 1,2,3 \ldots\ k$ of relation type arity $i (1 \leq i \leq k)$,where k is the maximum arity of relation type in T_R . Each relation type $r \in T_R^i$ of arity i has a signature $\sigma(r) = \prod_{i\ times} T_C$ specifying the maximum concept type of each of its arguments.
- I is the set of countable set of individual markers used to refer specific concepts.
- $*$ is the generic marker used to refer to an unspecified concept of a specify type.

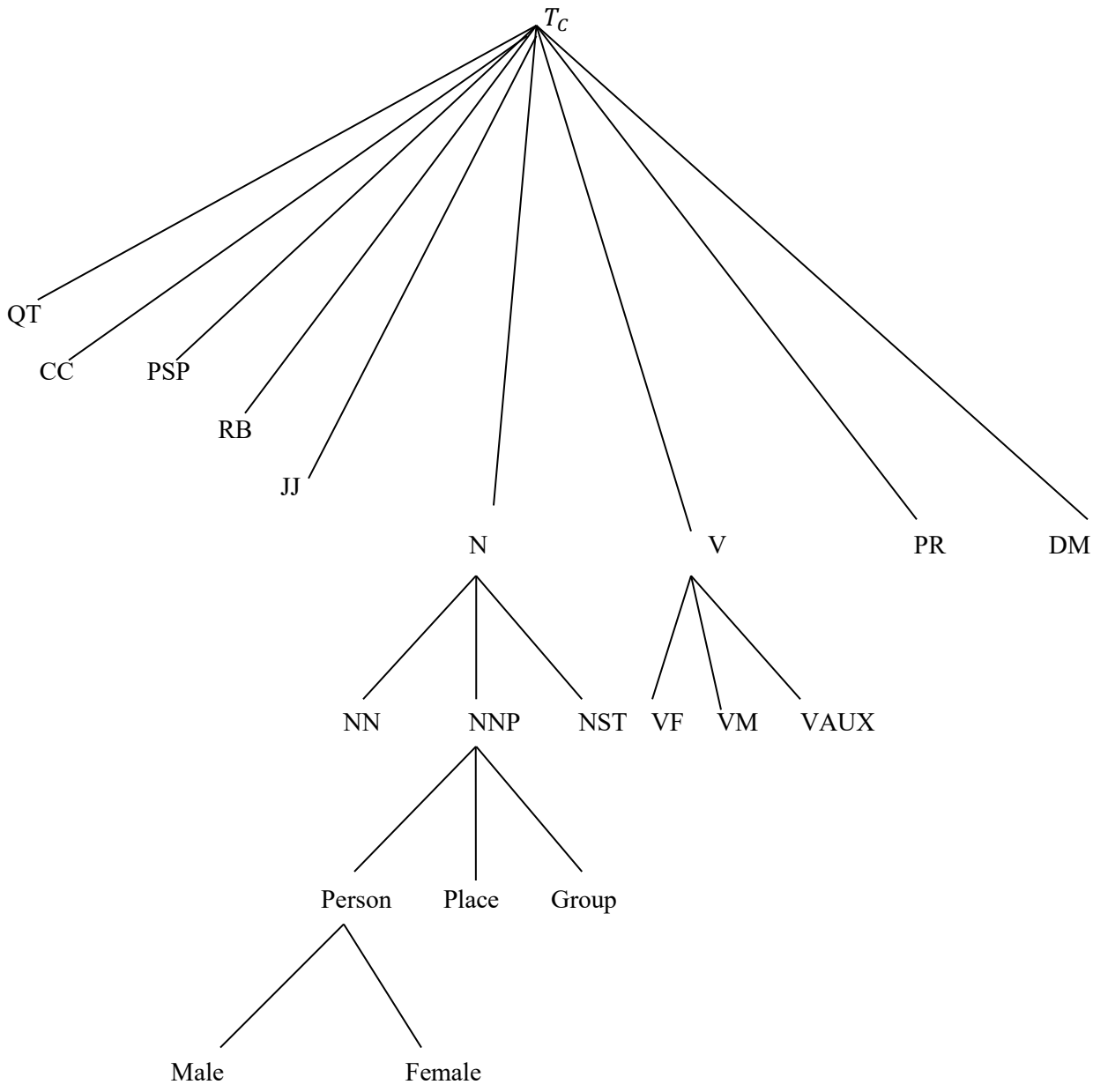


FIGURE 7.7(a): Concept type Hierarchy

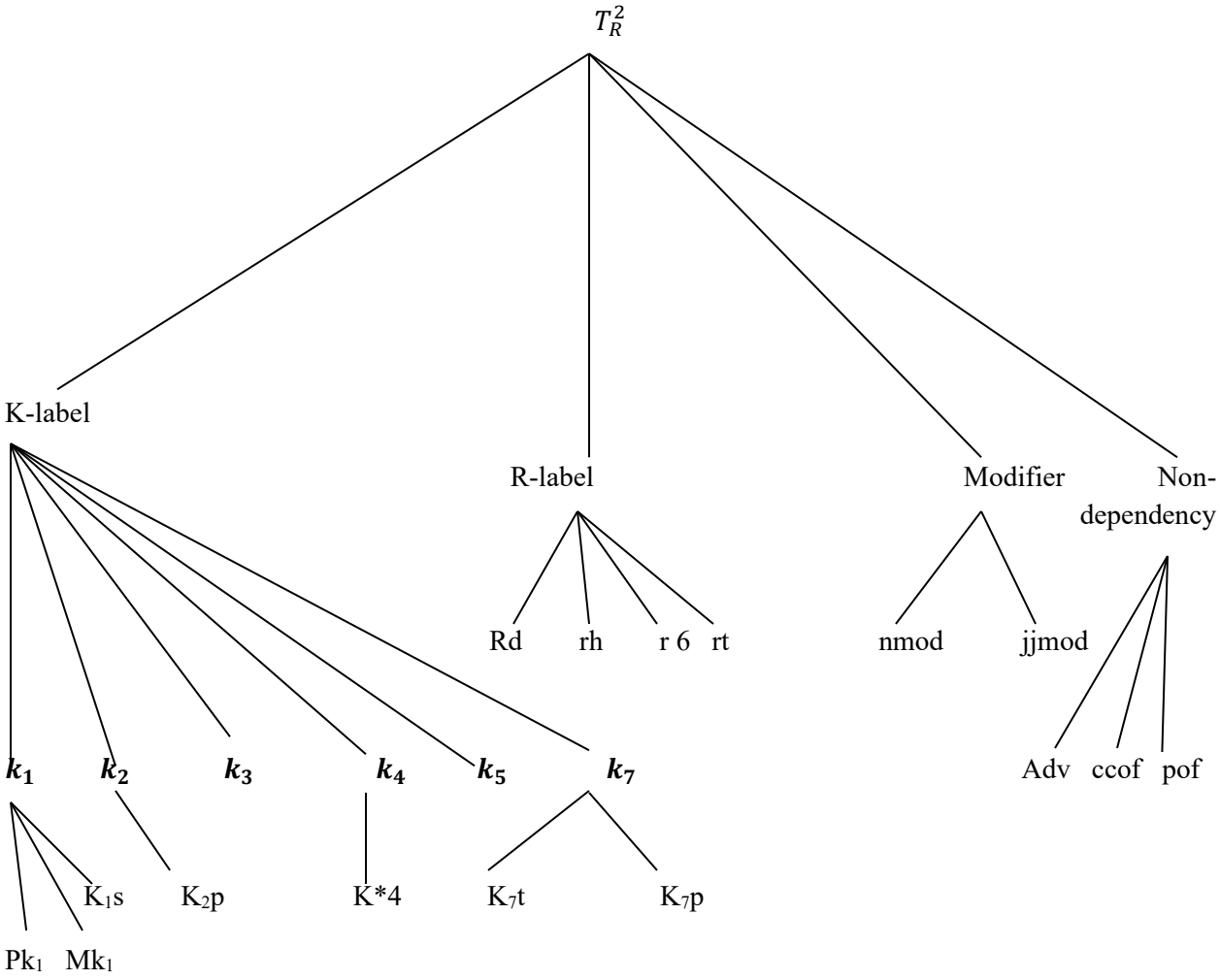


FIGURE 7.7 (b): Relation Type Hierarchy

7.3 The Constraint Graph:

A Bipartite Graph G is defined as $G = (V_C, V_R, E)$ where V_C, V_R are the set of nodes such that $V_C \cap V_R = \emptyset$ and E is the set of edges between $V_C$and V_R. To construct a problem bipartite graph for given sentence, first construct a constraint graph following the Paninian Framework [23] and then consider following the five steps:

1. For every source node s in the constraint graph, form a node $s$in V_C
2. For every demand node d in the constraint graph and every mandatory karaka k in the karaka chart for d, from a node v in V_R (thus for every pair (d, k) there is a node in V_R
3. For every demand node d in the constraint graph and every possible optional karaka $k$in the karaka chart for$d$, from a node v inV_R.
4. For every edge (d, s) labeled by karaka in the constraint graph create edge between nodes (d, k) in V_R to s in V_C .
5. If for a source node s in the constraint graph there is no mandatory karaka in the karaka chart for$d$, then optional karaka on the priority basis will be considered as mandatory.

Definition 7.3.1: Planar Graph

A planar graph is a graph that can be embedded in the plane, i.e., it can be drawn on the plane in such a way that its edges intersect only at their endpoints. In other words, it can be drawn in such a way that no edges cross each other [14].

Now, how to parse the projective sentences. The problem is identified that whether they are projective or not. To do so, under which condition a given Complete Bipartite graph of a sentences shows the projectivity. Here explore the Planarity of a graph, which defines the Projectivity in the frame of the following theorem.

Theorem 7.1:

Single Verbal Sentence is always Projective and the corresponding Complete Bipartite graph is always Planar.

Proof: For the Complete Bipartite graph the notation is $K_{m,n}$.

Here m is the verbal corpus and n is the non-verbal corpus(text)

The graph $K_{1,n}$ is planar $\forall\ n$ .It represent a Star Graph

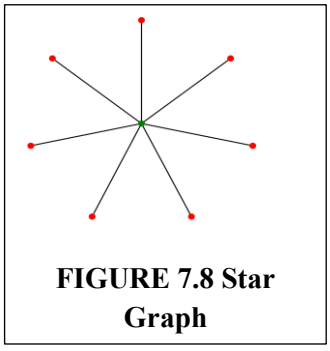


FIGURE 7.8 Star Graph

To show that $K_{1,n}$ is Projective. For the projectivity definition discuss in Chapter-5, Section5.3.1
If node $x_n$depends on node x_1 , then all nodes between x_1 andx_n are also subordinate tox_1 (i.e., dominated by x_1)

$$x_1 \rightarrow x_n \Rightarrow x_1 \rightarrow * x_j$$

$$\forall\, x_j \in X : (x_1 < x_j < x_n \vee\ x_1 > x_j > x_n)$$

$\rightarrow *$ is used to denote the reflexive and transitive closure of the relation.
Here $x_1 to\ x_n$ considering as a statement of the words.
Any statement that has only one verbal given is always Projective.
Hence the Proof.

Theorem 7.2: A Complete Bipartite graph $K_{m,n}$ for m=2 is planar and the corresponding sentence is projective if the conjunction ("AND", "OR") exist in the sentence and its position is just after the first verb i.e., if the position of the first verb is *n,* then the position of the conjunct should be *(n+1)*.

Proof: By applying the syntactic rule of Conjunction for Gujrati grammar, whenever the conjunction (POS: CCD) appears after the first verb the sentence can be divided in two independent sentences hence the corresponding $K_{2,\,n}$ can be partitioned in to two mutually exclusive graphs $\mathbf{K_{1,p}}$ and $K_{1,q}$ such that $p + q = n$. Both of them being the star graphs by applying thm1, the projectivity is established.
Hence the Proof.
Note that in linguistic Gujarti corpus any sentence has no more than two verbs, if more than three verbal text are available it is out of the thesis.

7.4 Conceptual Graph for Parsing (New Terminology):

The conceptual graph for a given Gujarati sentences us define the New terminology for the following:

Definition 7.4.1: Ordered Bipartite Graph

It is defined as

$$B = (V_C, V_R, E_B, l)$$

Which is formed by first considering Problem Bipartite Graph $G = (V_C, V_R, E)$ defined as chapter 6 and then defining linear ordering $\forall w_i \in V_C$ on set of edges incident to w_i.where V_R represent the verb words paired with all the possible mandatory and optional karakas and V_C is the set of all non verb words. An OBG requires that for each node in one of the classes of the bipartition, its neighbour (belonging to other partition) to be ordered [49]. For every verb- karaka combination $(v_j k)$ for each non- verb words(w_i), we defined the labelling

$$l_i\colon E_B \to \{1,2 \dots \dots |V_R|\}$$

On the edges of B by

$$l_i[v_j k, w_i] \forall k \in T_R, \forall w_i \in V_C = \begin{cases} 1 & , \text{ if k is mandatory } \forall j \\ 2 , & \text{ if k is optinal } \forall j \\ 3 & , \text{ otherwise} \end{cases}$$

Where $i = 1,2, \dots \dots \dots |V_C|, j = 1,2, \dots \dots |V_R|, and\ l_i(\{v_j k, w_i\})$ is the index of the edges $\{v_j k, w_i\}$ in the above ordering of the edges incident in B to V_C .The label $l_i \forall i \in |V_C|$, is called the order labeling the edges of B .Now we have for each $v_j k \in V_R, N_B^p(v_j k)$ denotes the p-th neighbor of $v_j k$, i.e, $w_p = N_B^p(v_j k)$,iff $\{v_j k, w_p\} \in E_B$.

Given a node $x_m \in V_C \cup V_R$, $\overline{N_B}(x_m)$ denotes the neighbors set of this node,

i.e., $\overline{N_B}(x_m) = \{u_m \in V_C \cup V_R | \{x_m, u_m\} \in E_B\}$.

Similarly, if$A \subseteq V_C \cup V_R$, its neighbours set is denoted as,

$$\overline{N_B}(x_m) = \cup_{x_m \in A} \overline{N_B}(x_m) - A.$$

Further assume that for each $w_i \in V_C$ there is $v_j k \in V_R\ and\ n \in N$ such that $w_i = N_B^n(v_j k)$;

i.e., B has no isolated vertices. OBG are appropriate tools to represent and visualize (directed) hyper graphs [33]. Visually, an ordered bipartite graph can be represented using boxes for vertices in V_C , ovals for vertices in V_R and integer labeled simple curves (edges) connecting boxes and ovals [33].

Definition 7.4.2: NL Support

$NLS = (T_C, T_R, I, *)$is a NL support related with the syntax and semantics of the natural language where:

T_C is a finite, partially ordered set of concept types which is depicted in Fig.7.7

T_R is a finite poset of relation types of arity 2 (T_R^2) and T_R^* (set of generic relation).
I is the set of countable set of individual markers, used to refer specific concepts and* is the generic marker used to refer unspecified concept.

Definition 7.4.3: NL labelling

λ as the labeling of all the vertices (words) of B by the elements of NLS as: $\forall\, c \in V_C$, $\lambda(c) \in T_C \times I \cup \{*\}$ and to avoid the ambiguity of relation node we only consider such $r \in V_R$, such that $\lambda(r) \in T_R^2 \cup T_R^*$ iff $l_i(r, w_i) = 1$.

$\lambda(r)$ $a$re determined by the various levels of relation arity tree, where r is the relationship between the words in the context of the given string and $\lambda(c)$ are determined by the levels of concept type hierarchy tree, where c is a word of given string. Finally, CG representing a parse tree of a given string (sentences) as $CG = (B, NLS, \lambda)$.

7.5 Concept Type Hierarchy and Relation arity:

Conceptual graphs may assert episodic information about particular individuals, or they may express general principles in the semantic network [93]. According to John F. Sowa [93] any representation must satisfy the following constraints:

- Connectivity: The algorithms for language parsing, generation, and reasoning depend on the ability to start from any concept and traverse the entire graph. Implementation must support some sort of forward and backward pointers linking all the nodes".
- "Generality: Although most primitive conceptual relations are dyadic, the formalism allows relations with any number of arcs. Furthermore, any concept may have any number of relations attached to it, and the number may increase as more assertions are made. The implementation must support of these options".
- "No privileged nodes: Any concept during a conceptual graph could also be treated as the head. The Choice of concept to express as a subject or predicate depends on focus and emphasis, but the representation shouldn't presuppose one choice of root or head (as trees and frames typically do)".

- "Canonical formation rules: The four rules of copy, restrict, join, and simplify are used throughout the system in reasoning and parsing. The implementation must make these operations fast and simple".
- Concept type nodes are the central directories for semantic attribute about a concept type. A part of speech for every such nodes, as in conceptual dependency and dependency parsing, the verb plays a central role within the structure. Relation type records specify semantic information a few conceptual relation types i.e., they specify the inter-relationship between two concept type nodes. In our context all the relation types are of arity two as each karaka can only links a demand node to a source node. Concept and Relation Hierarchy are depicted in Fig.7.7(a) & 7.7(b) respectively. T_C Consists of classes and sub-classes of a part of speech. In case of relation nodes are considering two types of relationship viz.
    - First, consider the karakas because the relation attribute between the concept nodes according to the karaka chart for each demand and source node with reference to the formal grammar rules.
    - Secondly, consider all kinds of generic relations which links two or more non-verb concept nodes. The first kind relation types are of arity two as each karaka can only links a requirement node to a source node.

7.6 Implementation:

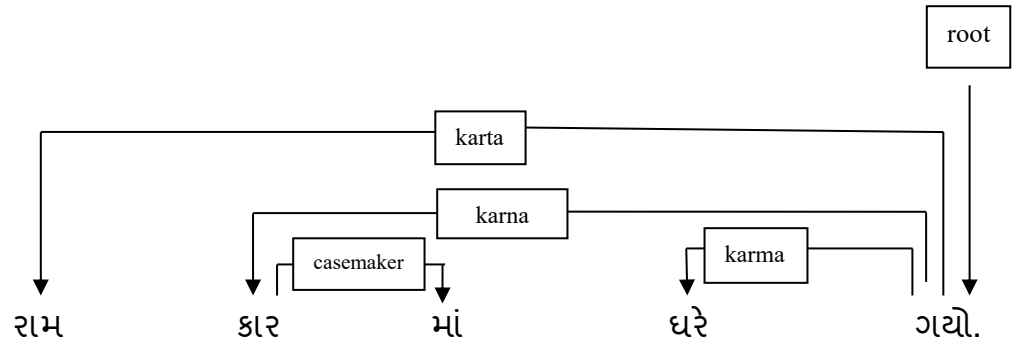


FIGURE 7.9: Linear Dependency

A conceptual graph as a parse tree for a given string (sentence). Sowa [93] introduced the CG describing a semantic interpreter that started with a parse tree then generating a CG representing the meaning of the sentence. In our study, describing a CG defined as $CG = (B, NLS, \lambda)$ as a parsing of the given sentence, with modified definition of λ to avoid the multiple relation between the same two words for this purpose, consider a projective sentence "રામ કાર માં ઘરે ગયો".

Construct a problem bipartite graph as mention in section 7.2 and then obtain an ordered bipartite graph (definition7.4.1) Members or elements of the NLS is from the relation arity tree (Fig.7.7b) and the Concept Type Hierarchy tree (Fig.7.7a). To identify the generic and individual marker consider the syntax and the semantics of the natural language. Finally applying λ to B, to get a CG as a parsing of the given sentence as depicted in Fig.7.11. The conceptual graph in Fig.7.9 is a sorted version of logic representing a parse. Each of the four concepts have a type label, which refers to: રામ, કાર માં, ઘરે and ગયો. One of the concepts is an individual marker. Each of the three conceptual relations has a type label that represents the type of relation: karta (k_1), destination (k_{2p}), or karana (k_3). The CG as a whole demonstrates that the person રામ is the agent of some instance of ગયો, ઘરે is the destination, and કાર માં is the instrument. Fig.7.11 can be translated to the Sowa 's formula as [92]:

$((\exists x)(\exists y)$ ગયો $(x) \wedge$ person (રામ) $\wedge$ ઘરે $(y) \wedge k_1\ (x,$ રામ$) \wedge k_{2p}\ (x,$ ઘરે$) \wedge k_3(x, y))$

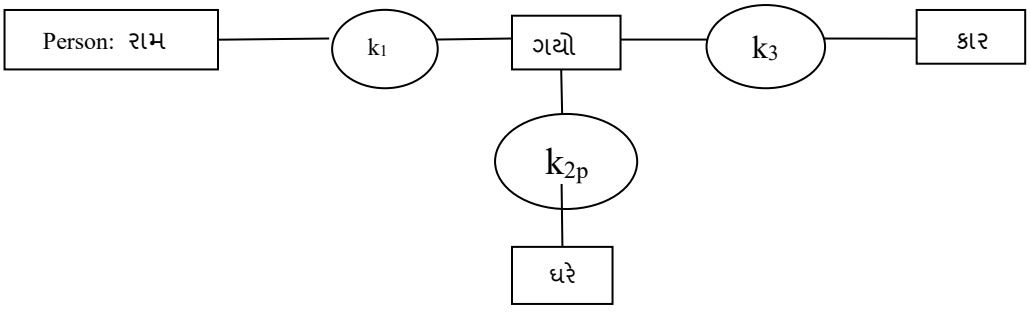


FIGURE 7.10: Conceptual Graph as a Parsing of the given sentence.

TABLE 7.1: Verb frame in the context of the given sentence

Arc label	Necessity	Vibhakti	Concept type
k_1	**m**	**0**	**n**
k_{2p}	**m**	**0**	**n**
k_3	**m**	માં	**n**

Theorem 7.3

Let $S_1, S_2 \subseteq X$, such that $X = S_1 \cup S_2$ where X is a non-projective string (sentence) and S_1 and S_2 are projective strings, if ∃ conceptual graphs CG_1 , CG_2 representing parsing S_1 and S_2 respectively, then there exists a conceptual graph CG^* representing a parsing of X such that $CG^* = CG_1 \cup CG_2 \cup e$, where e is a connector relation node between $CG_1$and CG_2.

Proof:

Non-projectivity structure, in contrast to projective dependency are common in all-natural language. Similar to that of Hindi non-projectivity is caused in Gujarati, due to various linguistic anomaly in the language, such as relative constructions, paired connectives, complex coordinating structures, interventions in verbal arguments by non-verbal modifiers, shared arguments in non-finite clauses, movement of modifiers, presence of case marker etc.

To check the validity of the theorem we consider a non−projective sentence S: “જે છોકરો ત્યાં બેઠો છે તે મારો ભાઈ છે’’ meaning “the boy who is sitting there he is my brother.” To analyze the given sentence we first break it into possible chunks as “જે છોકરો”, “ત્યાં”, “બેઠો છે”, “તે”, “મારો ભાઈ” and “છે”. The linear dependency and verb frame in the context of the sentence can give as follows:

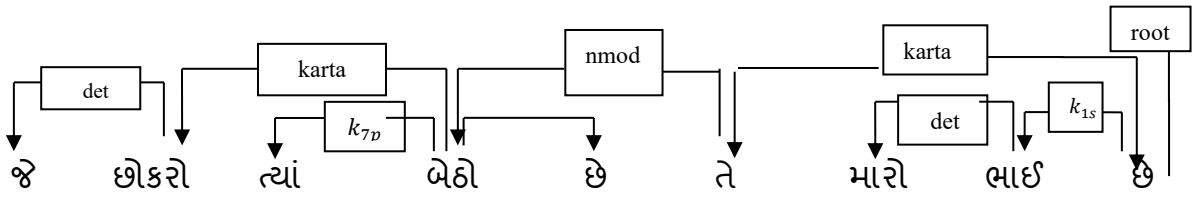


FIGURE 7.11: Linear Dependency non-projective

The sentence S can be decompose into two independent projective sentences as S_1 = "જે છોકરો ત્યાં બેઠો છે" and S_2 = "તે મારો ભાઈ છે''. After constructing the corresponding problem bipartite graphs for both sentences applying the definitions 7.4.1, 7.4.2, and 7.4.3. So, to get the Conceptual graphs $CG_1$and CG_2 as the parsing of the sentences S_1 and S_2 respectively. The basic requirement to construct a CG^* of a non projective sentence from conceptual graphs of two projective sub-sentences of the given sentence is a modifier connector, i.e., e . For S $e = nmod$, which is a noun modifier. Therefore, combining the $CG_1$and CG_2 by $e = nmod$ we get the CG^* given by Fig. 7.14.

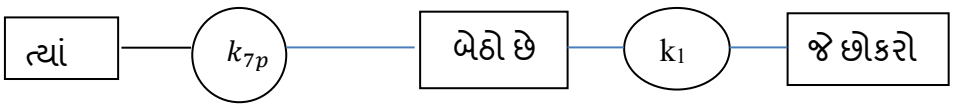


FIGURE 7.12: CG_1 for the sentence S_1

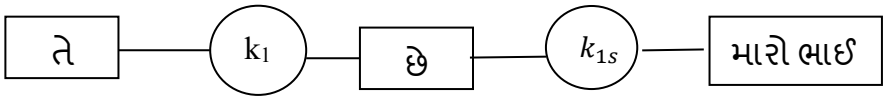


FIGURE 7.13: CG_2 for the sentence S_2

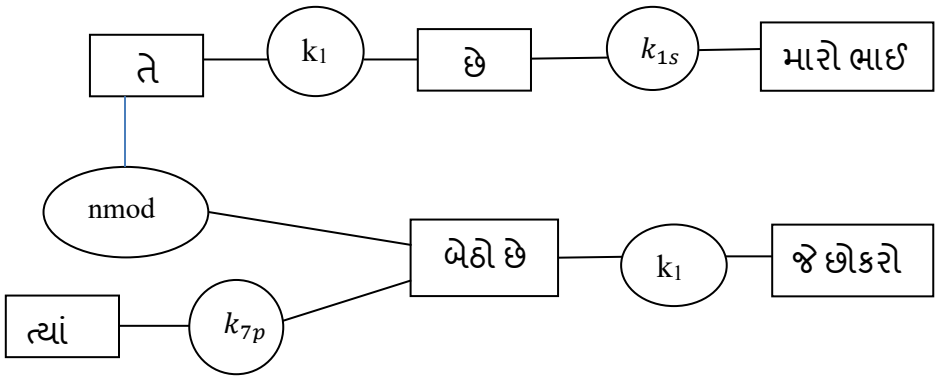


FIGURE 7.14: CG^* for the sentence S

TABLE 7.2: Verb frame for બેઠો છે in the context of the S

Arc label	Necessity	Vibhakti	Concept type
k_1	m	0	n
k7p	0	0	n

7.7 Parsing of Projective sentences (Implementation by Example)

માતા-પિતા માધવ ને શાળાએ મોકલે છે. (Eng.: Parents send Madhav to school.)

BG: Verb: મોકલે છે (send)

V_C : $\{NNP, NN\}$

V_R : $\{Verb\}$

Bipartite Graph: G:$\{V_C, V_R, E_G\}$

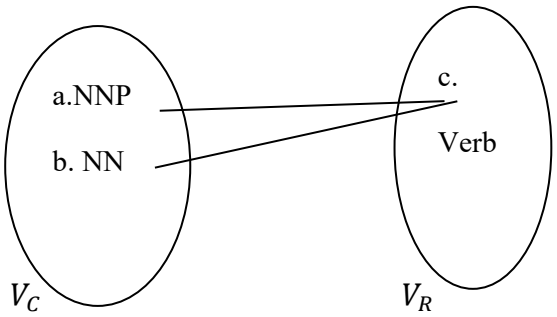


FIGURE 7.15: Bipartite Graph

e_1: $\{a, c\} e_2$: $\{b, c\}$

Ordered Bipartite Graph: (V_C, V_R, N_G)

$N_G: V_R \rightarrow V_C^+ isaMapping$

N_G(Verb):$\{NNP, NN\}$

$r \in V_R$,$N_G^i(r)$: $i^{th} neighbour of r.$

i.e.,$v = N_G^i(r), iff\{r, v\} \in E_G \& \ \ l(r, v) = i$

i.e., $l(e_1) = 1$, $l(e_2) = 2$

NL-Support:$(T_C, T_R, I, *)$

T_C : Specific structure, such as a tree, a lattice or a semi lattice.

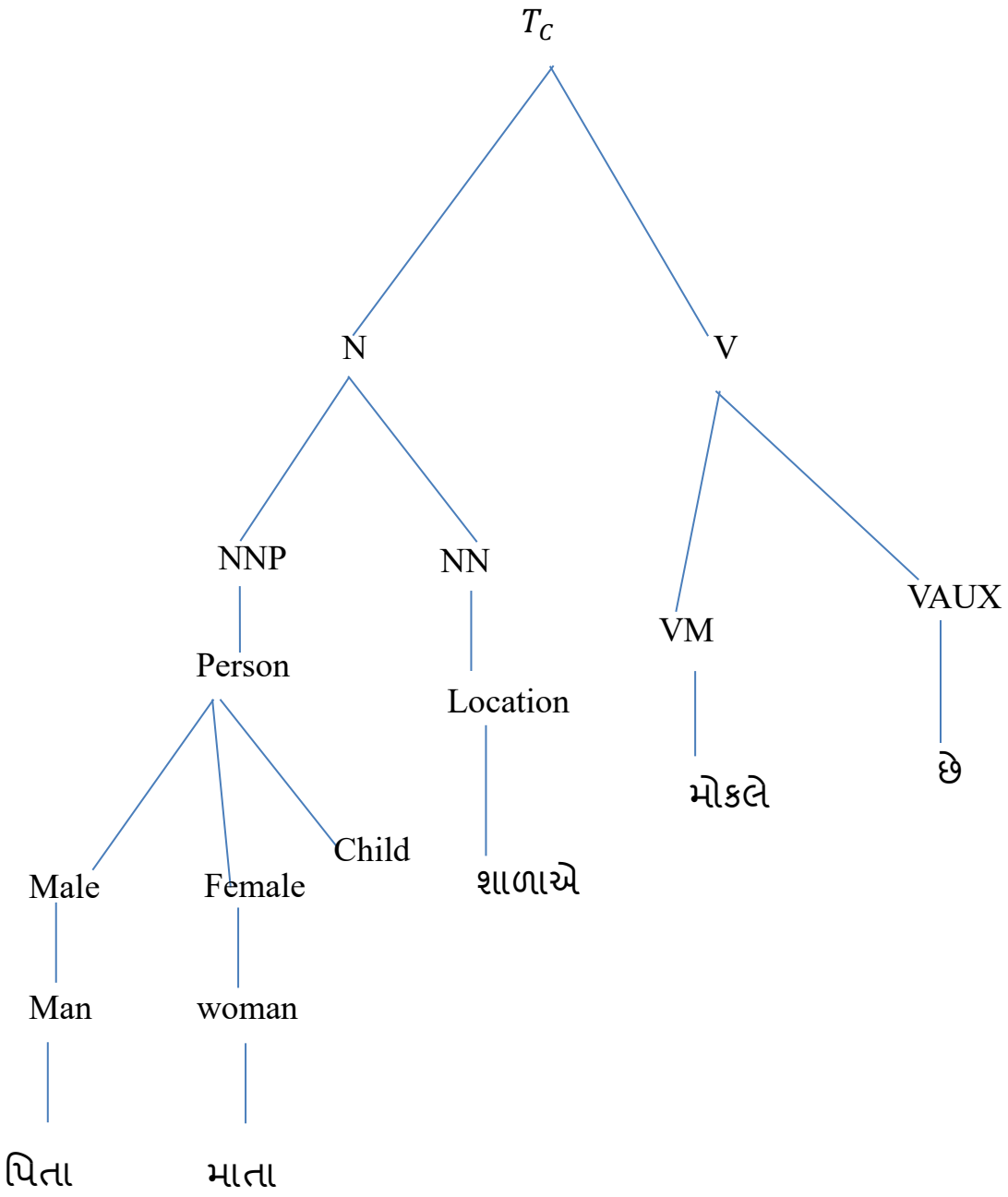


FIGURE 7.16: Concept Type Hierarchy

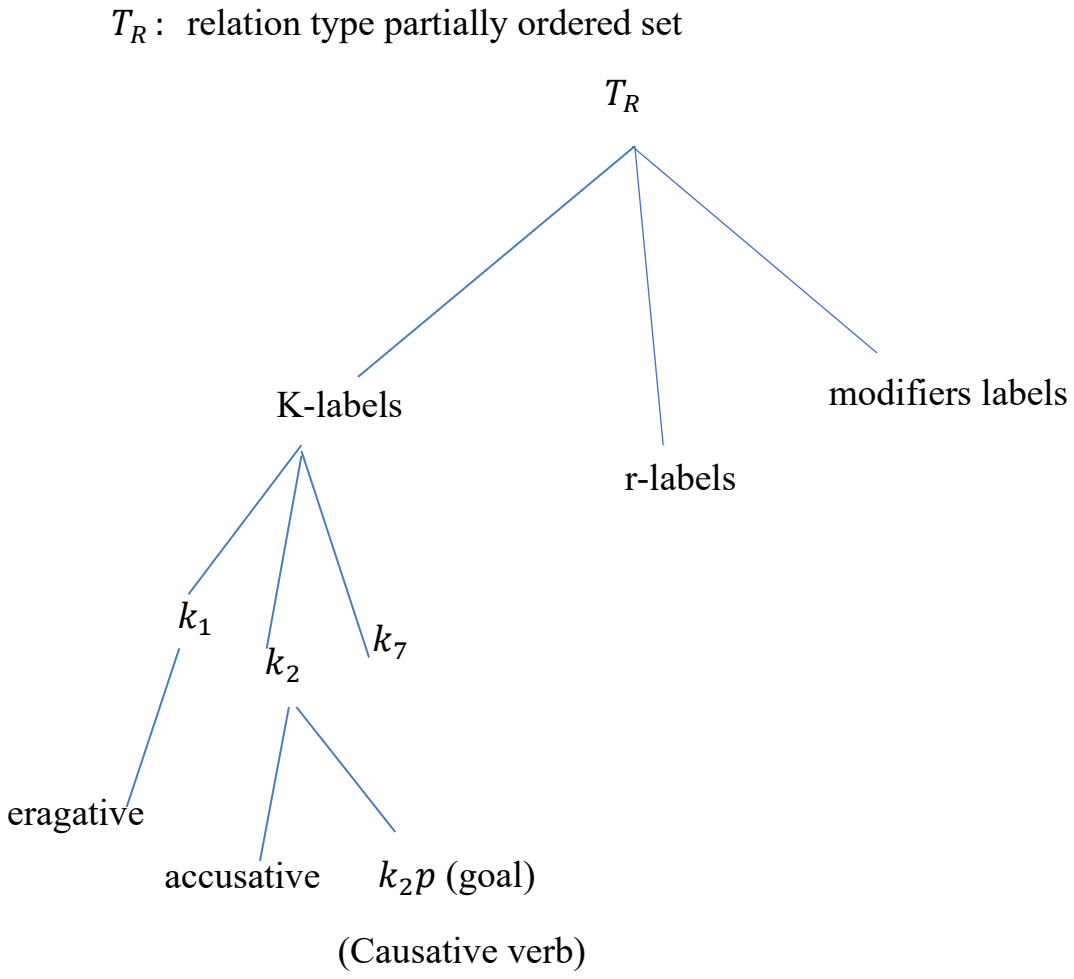


FIGURE 7.17: Relation Type Hierarchy

T_R : Relation type partially ordered set ≤.

T_R is Partitioned into subset $T_R^1, T_R^2, \ldots \ldots \ldots T_R^k$ of relation type arity 1……k respectively. (k≤ 1)

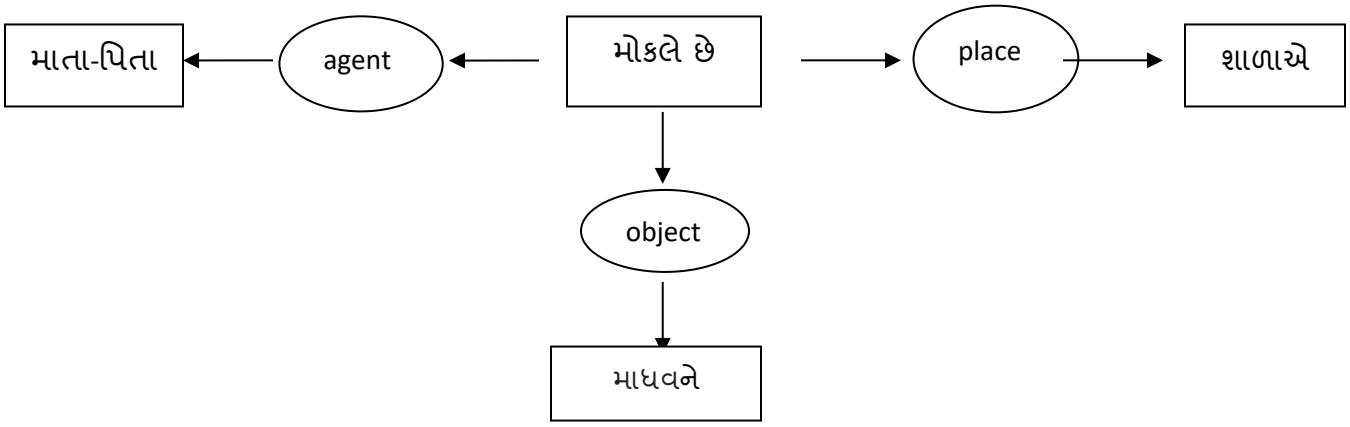


FIGURE 7.18 CG graph For the Sentence

Labelling: λ **:** $\boldsymbol{G \rightarrow S}$

i.e. $\forall \boldsymbol{r} \in \boldsymbol{V_R}$,λ $(\boldsymbol{r}) \in \boldsymbol{T_R^{dG(r)}}$, ,$\lambda$ $(\boldsymbol{r}) = \boldsymbol{t_r}$

$\forall \boldsymbol{c} \in \boldsymbol{V_C}$,λ $(\boldsymbol{c}) \in \boldsymbol{T_C} * (\boldsymbol{I} \cup \{*\})$, ,λ $(\boldsymbol{c}) = \big(\boldsymbol{type}(\boldsymbol{c}), \boldsymbol{marker}(\boldsymbol{c})\big)$,

$\boldsymbol{r} \in \boldsymbol{V_R}$ $\quad \boldsymbol{i,e.Verb} \in \boldsymbol{V_R}$ $\lambda(\mathbf{verb}) \in \boldsymbol{T_R^2}$,

$\lambda(\mathbf{verb}) = \mathbf{type}(\mathbf{r})$ =મોકલેછે

$\boldsymbol{c_1} \in \boldsymbol{V_C}\lambda$ **(** $\boldsymbol{c_1}$ **)** ∈માતા-પિતા / માધવ×**(I),**

$\boldsymbol{c_2} \in \boldsymbol{V_C}\lambda$**(b)** ∈શાળાએ×**(I)**

TABLE 7.3: Verb Frame મોકલે છે [Eng.: send]

Arc-label	rule	necessity	vibhakti	Lexical type
k_1	ergative	m	0	n
k_2	accusative	m	ને	n
k_2p	goal	m	0	n

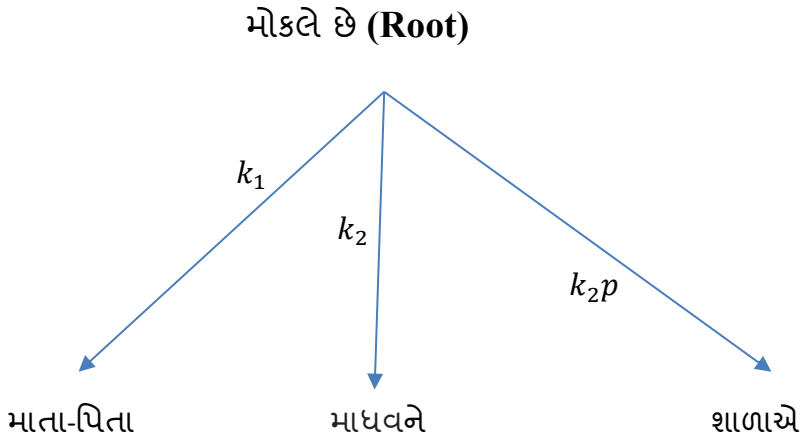


FIGURE 7.19: CG Paring for the Given Sentence

7.8 Conclusion:

A CG model to the parsing of Gujarati languages for the projective sentences and Non-Projective sentences. Using CG approach for parsing of a natural language has many advantages over other existing methods, the existence of a NL support distinguishes and sorts the hierarchy of concepts and relationship between the words. Here re-defines ordered bipartite graph, NL support in a new context of parsing. Conceptual Graph approach is a novel approach employed for Natural language parsing. It gives structured approach to the approach of Maximum Bipartite graph (discussed in chapter 6) by defining labels, defining preference or order etc. The derived mathematical properties could assist future work in research and the development of knowledge representation, in particular, in the area of parsing, which has many applications in Natural Language Processing. Finally propose a theorem to construct a CG as a parse of a non-projective sentence by decomposing it into projective sub parts. This theorem gives the condition based on planarity under which the user can identify whether the given sentence is projective or non-projective.

CHAPTER-8

Conclusion and Future Scope

8.1 Conclusion:

In the present experimental study, the development of Gujarati words and its pod tag sets. Experiments are carried out for the Viterbi algorithm for hidden markov model taggers and used to mat lab code. Based on the hidden markov model transition probabilities and Emission Probabilities matrix can be used. To finding the optimal sequences of tags applying the pseudo code for the Viterbi algorithm. The experimental result shows that the error carried out the mismatches which is highlighted with the same category. For example, the tag VM is original tag whereas Viterbi algorithm gives VAUX as predicted tag, which are the same category.

Experiment of the machine learning approaches for the Guajarati language using support vector machine and wavelet based neural network. Support Vector machine can perform their learning with all combination of given feature without increasing the complexity of the kernel function. Wavelets features are small time support to zoom in details like delta function. Their ability to adapt their scale and time support enables to frequency domain. New techniques applying to find the feature extraction for the support vector machine and the results are comparing with the viterbi algorithm for multi class problem. The Viterbi algorithm gives the best accuracy to compare the support vector machine. Neural networks are systems that are deliberately constructed to make use of some organization principles resembling those of human brain. They represent the promising new generation of information processing system. Neural networks are good at tasks such as

pattern matching and classification, function approximation, optimization. While traditional computers, because of their architecture, are efficient at these tasks, especially pattern matching tasks. Wavelet neural networks try to combine these aspects of wavelet transformation for the purpose of feature extraction and selection with characteristic decision capabilities of neural networks approaches. The WNN is constructed on the basis of wavelet transformation theory and is an alternative to feed forward neural networks for pattern classification. WNN are introduced a special feed forward neural network. Wavelet decomposition is a powerful tool for nonstationary signal. Wavelet neural networks (WNN) and support vector machine (SVM) are two advanced methods which are fit for classification. A comparative analysis of the two methods was conducted based on Gujarati data set. The results show SVM is much superior to WNN for small sample learning.

Non-projectivity for the Gujarati sentences and idea of the planarity graph established. The knowledge of non-projectivity classes used the performance of the Parser. The parser can have specialized machinery to handle non-projectivity only linguistic cues belonging to those classes are active. A parser can briefly describe as a tool to extract the relationship among the words of sentences. It can be solved with the help of Linear programming problem, bipartite matching problem etc. It makes use of two component a parser which is computer program and grammar which is declarative. It uses of karaka relation between verbs and nouns in sentences.

Mathematical techniques play a vital role to understand and solve the critical real-life problems. Chapter 6 demonstrates the simplest mathematical technique based on graph theory and linear programming problem. Bipartite graph generated the corresponding parsing can be obtained with the help if maximal matching subgraphs. LPP is one of the powerful techniques for parsing in NLP.A parse can be obtained from the constraint graph using simplex method. Verb frames are developed based on simple present tense and indicate habitual acts taking it as default. Karaka relation and the postpositions in the verb frame reflect the behaviour of the verb when it occurs simple present tense. This is done to bring in consistency while forming the various verb frames in Gujarati, the postposition of an argument might change with the TAM (tense, aspect and modality) information of the verb.

A very new technique is used for parsing name as Conceptual graph. Conceptual graphs are a visual knowledge representation formalism where information is encoded using bipartite graph.

This work is motivated by the belief that efficient graph theoretical techniques must be employed to enable the practical usability of mechanism.

8.2 Future Scope

Artificial Neural Network Or deep learning Parsing for the Gujarati Sentences. Deep learning refers to those artificial neural networks that are composed of the many layers. It is the quickest developing field in machine learning. It utilizes many-layered Deep Neural Networks to learn levels of representation and abstraction that make sense of data such as sound, picture, and content. Essential level, deep learning may be a machine learning technique. It encourages to a computer to filter inputs through layers to learn how to predict and classify formation. Perception is frequently the kind of pictures, content, or sound. The motivation for deep learning is that the manner that the human brain filters information.

A machine learning workflow starts being manually extracted from images with relevant features. The features are then wont to create a model that categorizes the objects within the image. In a deep learning workflow, relevant features are automatically extracted from images. Deep learning performs end-to-end learning – where a network is given data and a task to perform, like classification, and it learns the way to do that automatically. A key advantage of deep learning networks is that they often still improve because the size of your data increases.

Non- projectivity structure, in contrast to projectivity dependency are common in all-natural language. Mathematical equation is assisting to future work in research and development of knowledge representation in the area of parsing. Large scale efforts underway to create dependency treebank and parsers for Gujarati and other Indian Languages. As the dependency tree bank grows, more types of non- projective constructions that could bring interesting phenomenon.

APPENDIX – I

POS For Gujarati

Sr. No	Category		Tag Name	Example
	Top Level	**Sub type**		
1	**Noun**		**N**	
1.1		Common	NN	ચશ્માં, પેન
1.2		Proper	NNP	મોહન, રવિ.
1.3		Nloc	NST	ઉપર- નીચે
2.	**Pronoun**		**PR**	
2.1		Personal	PRP	હું ,તું ,તે.
2.2		Reflexive	PRF	પોતે, જાતે, સ્વયં.
2.3		Relative	PRL	જે, તે, જ્યાં,
2.4		Reciprocal	PRC	અરસ- પરસ,પરસ્પર
2.5		Wh- word	PRQ	કોણ,ક્યારે, ક્યાં
2.6		Indefinite		કોઈ ,કંઈક, કશુંક
3	**Demonstrative**		**DM**	
3.1		Deictic	DMD	આ
3.2		Relative	DMR	જે , જેને.
3.3		Wh-word	DMQ	કોણ,શું, કેમ

3.4		Indefinite		કોઈ ,કંઈક, કશુંક
4	**Verb**		**V**	
4.1		Main	VM	ખાશે, ખાધું.
4.2		Auxiliary	VAUX	છે, હતું કર્યું,
5	**Adjective**		**JJ**	
6	**Adverb**		**RB**	
7	**Postposition**		**PSP**	
8	**Conjunction**		**CC**	
8.1		Co-Ordinator	CCD	અને, કે
8.2		Sub-Ordinator	CCS	તેથી, એવું, કારણે
9	**Particles**		**RP**	
9.1		Default	RPD	પણ,જા, તો
9.2		Interjection	INJ	હે!! .અરે!!, ઓ!!
9.3		Intensifier	INTF	બહુ, ઘણું.
9.4		Negation	NEG	નહિ,ના.
10	**Quantifiers**		**QT**	
10.1		General	QTF	થોડું-ઘણું
10.2		Cardinals	QTC	એક-બે-ત્રણ
10.3		Ordinals	QTO	પહેલું, બીજું
11	**Residues**		**RD**	
11.1		Foreign word	RDF	tv
11.2		Symbol	SYM	$, *,&

11.3		Punctuation	PUNC	, : ; {} () .
11.4		Unknown	UNK	
11.5		Echo words	ECH	કામ-બામ, પાણી-બાણી

APPENDIX -II

Chunk Tag set (Gujarati)

Sr. No	Chunk Type	Tag Name	Example
1	Noun Chunk	NP	મારુ નવું ઘર_NP
2	Verb Chunk (finite)	VGF	મેં ઘરે ખાવાનું(ખાધુ_VM)_VGF
3	Verb Chunk (non-Finite)	VGNF	મેં (ખાતા ખાતા_VM)_VGNF tv જોયું
4	Verb Chunk	VG	મને રાતે (ખાવાનું_VM)_VGNN ગમે છે.
5	Adjectival Chunk	JJP	તે છોકરી(સુંદર _JJ) _JJP છે.
6	Adverb Chunk	RBP	તે (ધીરે ધીરે_RB)_RBP ચાલી રહ્યો હતો.
7	Conjuncts	CCP	(રામ_NP) (અને_CCP) (શ્યામ_NP)

APPENDIX-III

Dependency Tag set

No.	Tag Name	Tag Description	Labels
1	k1	કર્તા (Doer/ agent/subject)	K - Labels
2	k1s	સંજ્ઞા પૂરક (noun complement)	
3	k2	કર્મ (object /patient)	
4	k2p	ધ્યેય (Goal)	
5	k2s	કર્મ સમાધિકરણ (object compliment)	
6	k3	કર્ણ (instrument)	
7	k4	સંપ્રદાન (recipient)	
8	k*4	સમાનતા (similarity)	
9	k5	અપાદાન (Source)	
10	k7	અન્યત્ર સ્ત્રોત (location elsewhere)	
11	k7p	જગ્યા માં સ્થાન (location in space)	
12	k7t	સમય માં સ્થાન (location in time)	
13	pk1	કારકિર્દી (Causer)	
14	mk1	મધ્યસ્થ કર્તા (madhystha karta)	
15	jk1	પ્રાયોજિત કર્તા (Causee)	
16	ras	સહયોગી (associative)	R- Labels
17	rd	દિશા (Direction)	
18	rh	કારણસર (cause)	
19	r 6	માલિકીનું (possessive)	
20	rt	તાદર્થ્ય (purpose)	
21	nmod	નામ ના સંશોધકો(Noun modifier)	Modifier labels
22	jjmod	વિશિષ્ટ (adjective modifier)	

23	adv	ક્રિયા વિષેશણ (adverb/manner)	(Other) Non-dependences
24	ccof	સંબંધ સાથે જોડાયેલું (Conjunction)	
25	pof	સબંધ નો ભાગ(part of complex predicates)	
26	nmod-relc	પ્રકાર ના સંજ્ઞા સંશોધક (noun modifier of thr type)	
27	jjmod-relc	પ્રકારના વિશિષ્ટ સંશોધક(adjective modifier of the relative)	
28	rbmod-relc	Adverb modifier of the relative	

APPENDIX IV

Special Verb Frame

1. મીરા ને ખાવા નું બનાવતા આવડે છે. (Mira knows cooking.)

Arc -label	necessity	vibhakti	lextype
k_4a	m	ને	n
k_1	m	0	n

2. મજુર ટ્રક માં ગુણ ભરે છે. (worker load bags in the truck.)

Arc -label	necessity	vibhakti	lextype
k_1	m	0	n
k_7p	m	માં	n
k_2	m	0	n

3. આ બુક 125rs ની આવે છે. (The book cost Rs.125.)

Arc -label	necessity	vibhakti	lextype
k_1	m	0	n
k_7	m	ની	n

4. આ બુક 125rs માં મળે છે. (This book comes for Rs.125.)

Arc -label	necessity	vibhakti	lextype
k_1	m	0	n
r_6v	m	માં	n

5. હું રામ ની પાસે દરવાજો ખોલાવું છું (I cause Ram to open the door.)

Arc -label	necessity	vibhakti	lextype
pk_1	m	0	n
jk_1	m	ની પાસે	n
k_2	m	0	n

6. રામ દિલ્હી આવે છે. (ram comes to Delhi.)

Arc -label	necessity	vibhakti	lextype
k_1	m	0	n
k_2	m	0	n

7. રામ સીતા ને મળે છે. (Tam meets with Sita) (here Sita is an associative participant)

Arc -label	necessity	vibhakti	lextype
k_1	m	0	n
ras	m	ને	n

8. મોહન રામ ને તેના મિત્ર થી લડાવે છે. (Mohan makes Ram to fight with his Friend)

Arc -label	necessity	vibhakti	lextype
pk_1	m	0	n
jk_1	m	ને	n
k_2	m	થી	n

9. રામ ફળ ખાઈ ને મોહન ને બોલાવે છે.(After eating fruits Ram calls to Mohan)

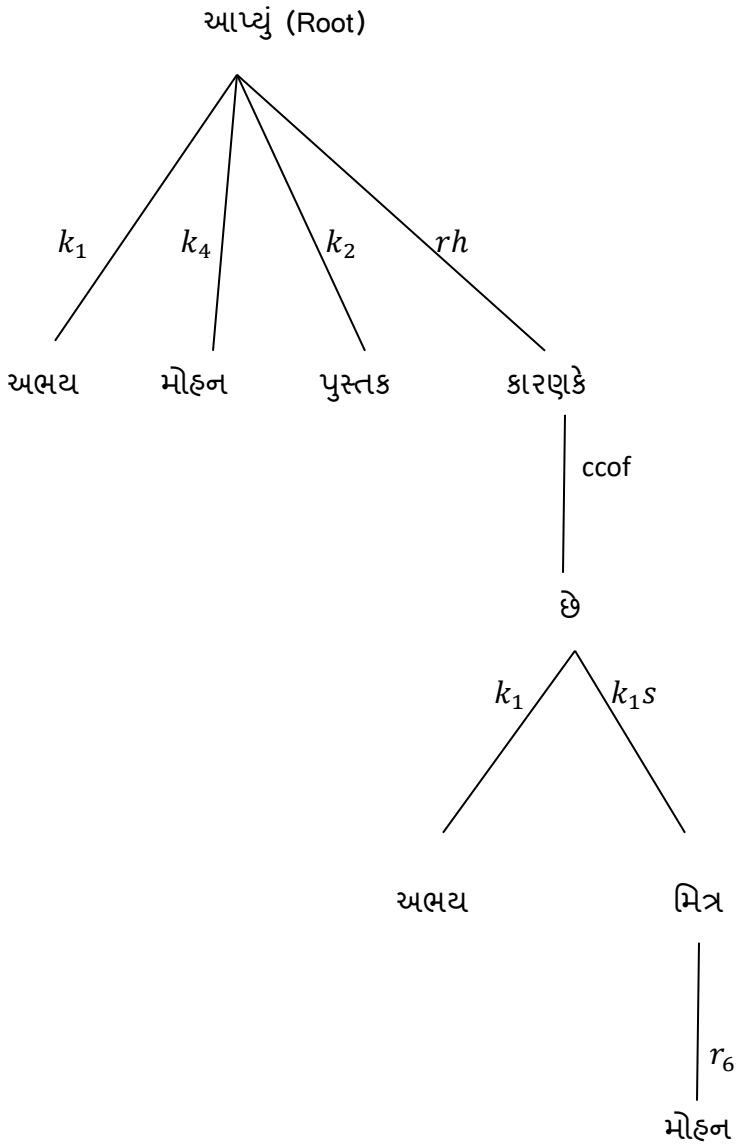
Arc -label	necessity	vibhakti	lextype
k_1	m	0	n
k_2	m	ને	n

APPENDIX V

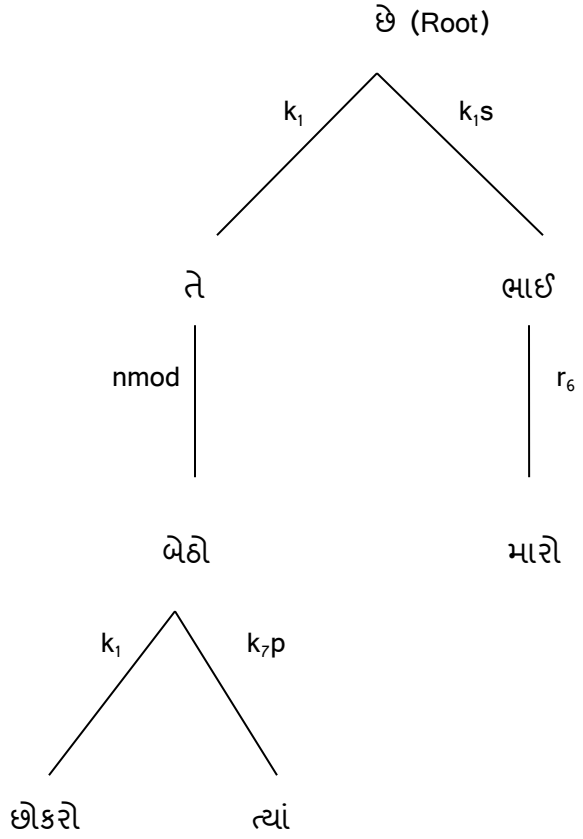
Sample Database

1.અભયે મોહનને સારું પુસ્તક આપ્યું કારણકે અભય મોહનનો મિત્ર છે.

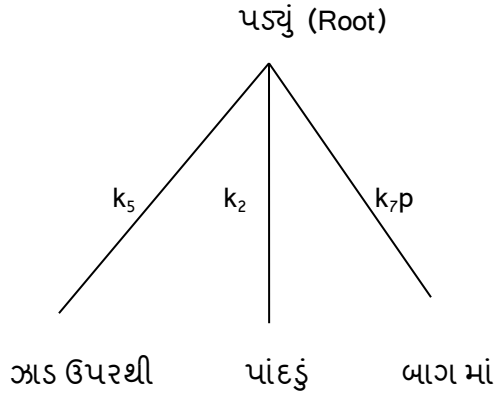
(Abhay gave Mohan a good book because Abhay is a good friend of Mohan.)



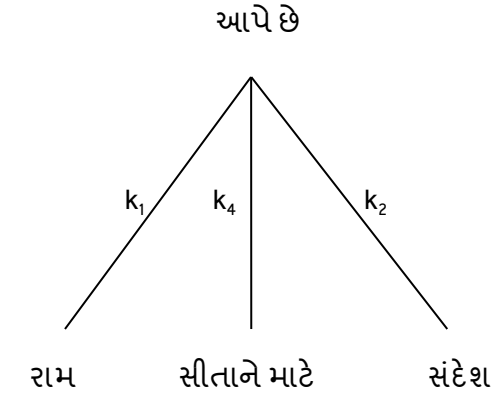
2. જે છોકરો ત્યાં બેઠો છે, તે મારો ભાઈ છે. (The Boy, who is sitting there, is my brother)



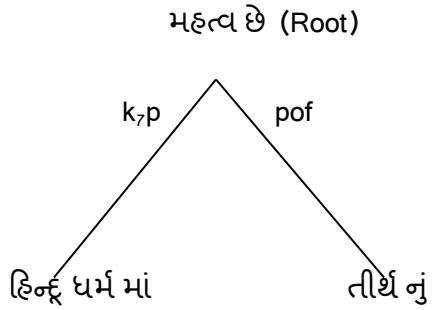
3. ઝાડ ઉપરથી બાગ માં પાંદડું પડ્યું. (Leaves fell from the tree in the garden)



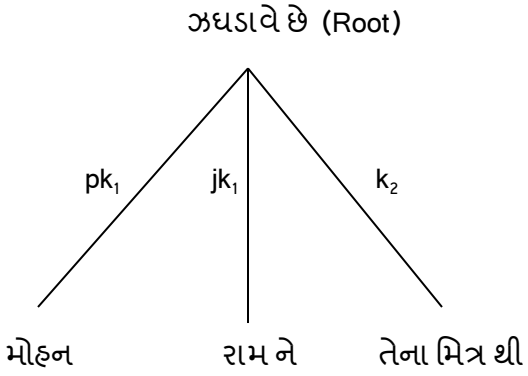
5. રામ સીતાને માટે સંદેશ આપે છે. (Ram sends message to Sita)



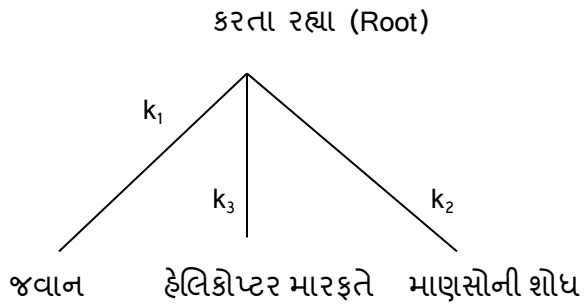
6. હિન્દૂ ધર્મ માં તીર્થ નું બહુ મહત્વ છે. (Tirtha is very important in Hinduism)



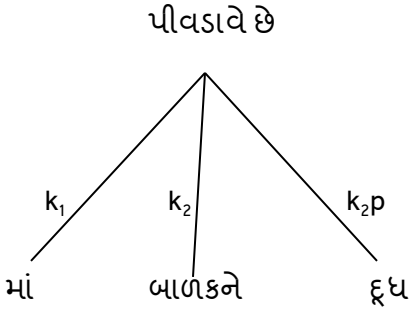
7. મોહન રામ ને તેના મિત્ર થી ઝઘડાવે છે. (Mohan makes Ram to fight with his friend.)



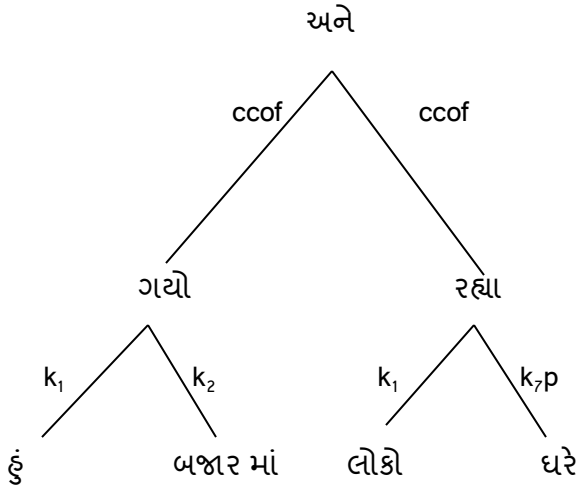
8. જવાન હેલિકોપ્ટર મારફતે માણસોની શોધ કરતા રહ્યા.(The men continued to search through the helicopter)



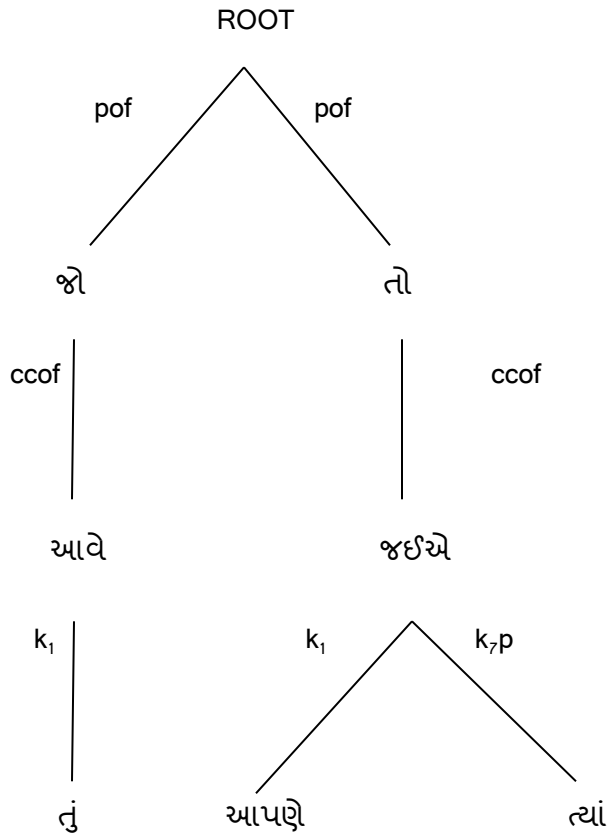
9. માં બાળકને દૂધ પીવડાવે છે. **(Breastfeed in baby)**



10. હું બજાર માં ગયો અને તે લોકો ઘરે રહ્યા. (I went to the Market and those people stayed at home)



11. જો તું આવે તો આપણે ત્યાં જઈએ. (Had you come, we would have gone there)



APPENDIX VI

Vibhakti (વિભક્તિ)

Def: વિભક્તિ શબ્દ વિ+ ભક્તિ પરથી બનેલો છે.' વિ' એટલે વિશેષ અને ' ભક્તિ' એટલે જોડાવવું.અર્થાત એક પદ સાથે બીજા પદ નું વિશષ પ્રકારનું ધનિષ્ઠ જોડાણ એટલે વિભક્તિ.

વિભક્તિ	પ્રત્યય	શું દર્શાવે છે?	નિયમ	ઉદાહરણ
કર્તા	એ	ક્રિયા નો કરનાર	ક્રિયાપદ ને કોણ-કોણે શબ્દ વડે પ્રશ્ન પૂછવામાં આવે છે	રામ વન માં જાય છે. રામે રાવણને માર્યો.
કર્મ	ને	ક્રિયા નું કર્મ	ક્રિયાપદ ને શું -કોને શબ્દ વડે પ્રશ્ન પૂછવામાં આવે છે	રોનક ધ્વનિ ને ચોકલેટ આપે છે
કર્ણ	થી ,થકી, વડે, દ્વારા	ક્રિયા નું સાધન	ક્રિયાપદ ને શાનાથી, શના થકી શાના વડે વગેરે શબ્દ વડે પ્રશ્ન પૂછવામાં આવે છે	મિહિર સાયકલ દ્વારા સ્કૂલ જાય છે. કિરણ બોલપેન થી લખે છે
સંપ્રદાન	માટે, વાસ્તે ,સારું , કાજે	ક્રિયા નો હેતુ, કારણ	ક્રિયાપદ ને કોને માટે ,કોને વાસ્તે શબ્દ વડે પ્રશ્ન પૂછવામાં આવે છે.	પીન્કી નિકિતા ને માટે ફૂલ લાવે છે
અપાદાન	માંથી, અંદરથી ,ઉપરથી	છૂટુંપાડવાની ક્રિયામાં સ્થિર	ક્રિયાપદ ને ક્યાંથી, કોની પાસેથી શબ્દ વડે પ્રશ્ન પૂછવામાં આવે છે	શ્રીકાંત કંપાસ માંથી પેન લે છે
સંબંધ	નો, ની, નું ,ના	સંબંધ દર્શાવનાર પદ	વ્યક્તિ- વ્યક્તિનો ,વ્યક્તિ-વસ્તુનો, વસ્તુ -વસ્તુનો	તે મારુ પુસ્તક છે.

			સંબંધ દર્શાવવા માટે	
અધિકારણ	માં, અંદર, ઉપર	ક્રિયાનું આધારવાચક પદ	ક્રિયાપદ ને ક્યાં-ક્યારે શબ્દ વડે પ્રશ્ન પૂછવામાં આવે છે	યાશીકા પલંગ પર બેઠી છે.
સંબોધન	હે, અરે, ઓ, અરેરે	જે ને સંબોધન થાય તે	જયારે કોઈ ને સંબોધન કરવું હોય ત્યારે ,સંબોધ પછી ઉદ્ગાર ચિન્હ અથવા અલ્પવિરામ મુકાય છે	હે ઈશ્વર! મારા પર કૃપા કરો. ઓ મિત્રો ! શાંતિ જાળવો

APPENDIX VII

Projective and Non-Projective Constraints using LPP

Projective Sentence-1

રામ સીતાને માટે સંદેશ મોકલે છે.

K1	K4	K2	M.V
a	b	c	A

Constraint C1: $M_{A,K1} : x_{A,K1,a} + x_{A,K1,c} = 1$ $M_{A,K2} : x_{A,K2,a} + x_{A,K2,c} = 1$

Constraint C2: $O_{A,K4} : x_{A,K4,b} \leq 1$

Constraint C3: $S_a : x_{A,K1,a} + x_{A,K2,a} = 1$ $S_b : x_{A,K4,b} = 1$ $S_c : x_{A,K1,c} + x_{A,K2,c} = 1$

Take $x_{A,K1,a} = y_1$, $x_{A,K1,c} = y_2$, $x_{A,K2,a} = y_3$, $x_{A,K2,c} = y_4$, $x_{A,K4,b} = y_5$

We get

$M_{A,K1} : y_1 + y_2 = 1$

$M_{A,K2} : y_3 + y_4 = 1$

$O_{A,K4} : y_5 \leq 1$

$S_a : y_1 + y_3 = 1$, $S_b : y_5 = 1$, $S_c : y_2 + y_4 = 1$

The cost function to be minimized is:

Min Z = $y_1 + y_2 + y_3 + y_4 + y_5$

Subject to,

$y_1 + y_2 = 1$

$y_3 + y_4 = 1$

$y_5 \leq 1$

$y_1 + y_3 = 1$

$y_5 = 1$

$y_2 + y_4 = 1$

Result: $y_1 = 1$, $y_4 = 1$, $y_5 = 1$ Parsing tree show on Page No. 146

Projective Sentence-2

રામે રાવણ ને તીરથી માર્યો.

K1	K2	K3	M.V.
a	b	c	A

Constraint C1: $M_{A,K1} : x_{A,K1,a} + x_{A,K1,b} = 1$ $M_{A,K2} : x_{A,K2,a} + x_{A,K2,b} = 1$

Constraint C2: $O_{A,K3} : x_{A,K3,c} \leq 1$

Constraint C3: $S_a : x_{A,K1,a} + x_{A,K2,a} = 1$ $S_b : x_{A,K1,b} + x_{A,K2,b} = 1$ $S_c : x_{A,K3,c} = 1$

Take $x_{A,K1,a} = y_1$, $x_{A,K1,b} = y_2$, $x_{A,K2,a} = y_3$, $x_{A,K2,b} = y_4$, $x_{A,K3,c} = y_5$

We get

$M_{A,K1} : y_1 + y_2 = 1$

$M_{A,K2} : y_3 + y_4 = 1$

$O_{A,K3} : y_5 \leq 1$

$S_a : y_1 + y_3 = 1$, $S_b : y_2 + y_4 = 1$, $S_c : y_5 = 1$

The cost function to be minimized is:

Min Z = $y_1 + y_2 + y_3 + y_4 + y_5$

Subject to,

$y_1 + y_2 = 1$

$y_3 + y_4 = 1$

$y_5 \leq 1$

$y_1 + y_3 = 1$

$y_5 = 1$

$y_2 + y_4 = 1$

Result: $y_1 = 1$, $y_4 = 1$, $y_5 = 1$ Parsing tree show on Page No. 9

Projective Sentence-3

માતા-પિતા બાળકોને શાળા એ મોકલે છે.

K1	K2	K7P	M.V.
a	b	c	A

Constraint C1: $M_{A,K1} : x_{A,K1,a} + x_{A,K1,b} = 1$ $M_{A,K2} : x_{A,K2,a} + x_{A,K2,b} = 1$

Constraint C2: $O_{A,K7P} : x_{A,K7P,c} \leq 1$

Constraint C3: $S_a : x_{A,K1,a} + x_{A,K2,a} = 1$ $S_b : x_{A,K1,b} + x_{A,K2,b} = 1$ $S_c : x_{A,K7P,c} = 1$

Take $x_{A,K1,a} = y_1$, $x_{A,K1,b} = y_2$, $x_{A,K2,a} = y_3$, $x_{A,K2,b} = y_4$, $x_{A,K7P,c} = y_5$

We get

$M_{A,K1} : y_1 + y_2 = 1$

$M_{A,K2} : y_3 + y_4 = 1$

$O_{A,K3} : y_5 \leq 1$

$S_a : y_1 + y_3 = 1$, $S_b : y_2 + y_4 = 1$, $S_c : y_5 = 1$

The cost function to be minimized is:

Min Z = $y_1 + y_2 + y_3 + y_4 + y_5$

Subject to,

$y_1 + y_2 = 1$
$y_3 + y_4 = 1$
$y_5 \leq 1$
$y_1 + y_3 = 1$
$y_5 = 1$
$y_2 + y_4 = 1$

Result: $y_1 = 1$, $y_4 = 1$, $y_5 = 1$ Same as Parsing tree show on Page No. 132

Projective Sentence-4

રામ સીતાને મળે છે.

K1	K2	M.V.
a	b	A

Constraint C1: $M_{A,K1} : x_{A,K1,a} + x_{A,K1,b} = 1$ $M_{A,K2} : x_{A,K2,a} + x_{A,K2,b} = 1$

Constraint C2: $O_{A,K3} : 0$

Constraint C3: $S_a : x_{A,K1,a} + x_{A,K2,a} = 1$ $S_b : x_{A,K1,b} + x_{A,K2,b} = 1$

Take $x_{A,K1,a} = y_1$, $x_{A,K1,b} = y_2$, $x_{A,K2,a} = y_3$, $x_{A,K2,b} = y_4$

We get

$M_{A,K1}: y_1 + y_2 = 1$

$M_{A,K2}: y_3 + y_4 = 1$

$S_a : y_1 + y_3 = 1, \quad S_b : y_2 + y_4 = 1$

The cost function to be minimized is:

Min Z = $y_1 + y_2 + y_3 + y_4$

Subject to,

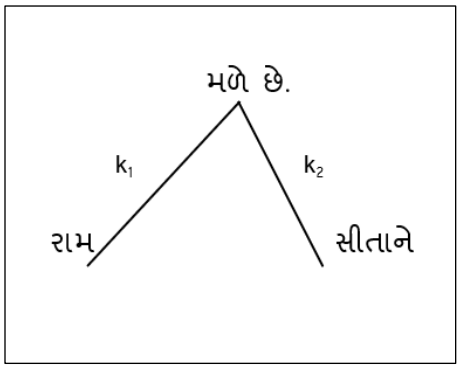
$y_1 + y_2 = 1$

$y_3 + y_4 = 1$

$y_1 + y_3 = 1$

$y_2 + y_4 = 1$

Result: $y_1 = 1$, $y_4 = 1$



Non-Projective Sentence-1

માણસે નાવ બનાવવામાટે લાકડી કાપી .

K1	K2	Verb.modi.	K2	M.V.
a	b	B	c	A

Constraint C1:

$M_{A,K1}: x_{A,K1,a} = 1$ $\quad M_{A,K2}: x_{A,K2,b} + x_{A,K2,c} = 1$, $M_{B,K2}: x_{B,K2,b} = 1$

Constraint C2: $O_{A,K4} : 0$

Constraint C3: $S_a : x_{A,K1,a} = 1$

$S_b : x_{A,K2,b} + x_{B,K2,b} = 1$

$S_c : x_{A,K2,c} = 1$

Take $x_{A,K1,a} = y_1$, $x_{A,K2,b} = y_2$, $x_{A,K2,c} = y_3$, $x_{B,K2,b} = y_4$

We get $M_{A,K1}: y_1 = 1$

$M_{A,K2}: y_2 + y_3 = 1$

$M_{B,K2}: y_4 = 1$

$S_a : y_1 = 1,$

$S_b : y_2 + y_4 = 1$,

$S_c : y_3 = 1$

The cost function to be minimized is:

Min Z = $y_1 + y_2 + y_3 + y_4$

Subject to,

$y_1 = 1$

$y_2 + y_3 = 1$

$y_4 = 1$

$y_3 = 1$

$y_2 + y_4 = 1$

Result: $y_1 = 1$, $y_3 = 1$, $y_4 = 1$ Same as Parsing tree show on Page No. 105

Non-Projective Sentence-2

રામ	ફળ	ખાઈ ને	મોહનને	બોલાવે છે
K1	K2	V.modi	k2	M.V.
a	b	B	c	A

Constraint C1:

$M_{A,K1} : x_{A,K1,a} = 1$ $M_{A,K2} : x_{A,K2,b} + x_{A,K2,c} = 1$, $M_{B,K2} : x_{B,K2,b} = 1$

Constraint C2:

$O_{A,K4} : x_{A,K4,c} \leq 1$

Constraint C3:

$S_a : x_{A,K1,a} = 1$

$S_b : x_{A,K2,b} + x_{B,K2,b} = 1$

$S_c : x_{A,K2,c} + x_{A,K4,c} = 1$

Take $x_{A,K1,a} = y_1$, $x_{A,K2,c} = y_2$, $x_{A,K2,b} = y_3$, $x_{B,K2,b} = y_4$, $x_{A,K4,c} = y_5$

We get

$M_{A,K1} : y_1 = 1$

$M_{A,K2} : y_2 + y_3 = 1$

$M_{B,K2} : y_4 = 1$

$O_{A,K4} : y_5 \leq 1$

$S_a : y_1 = 1,$

$S_b : y_3 + y_4 = 1$,

$S_c : y_2 + y_5 = 1$

The cost function to be minimized is:

Min Z = $y_1 + y_2 + y_3 + y_4 + y_5$

Subject to, $y_1 = 1$

$y_2 + y_3 = 1$

$y_4 = 1$

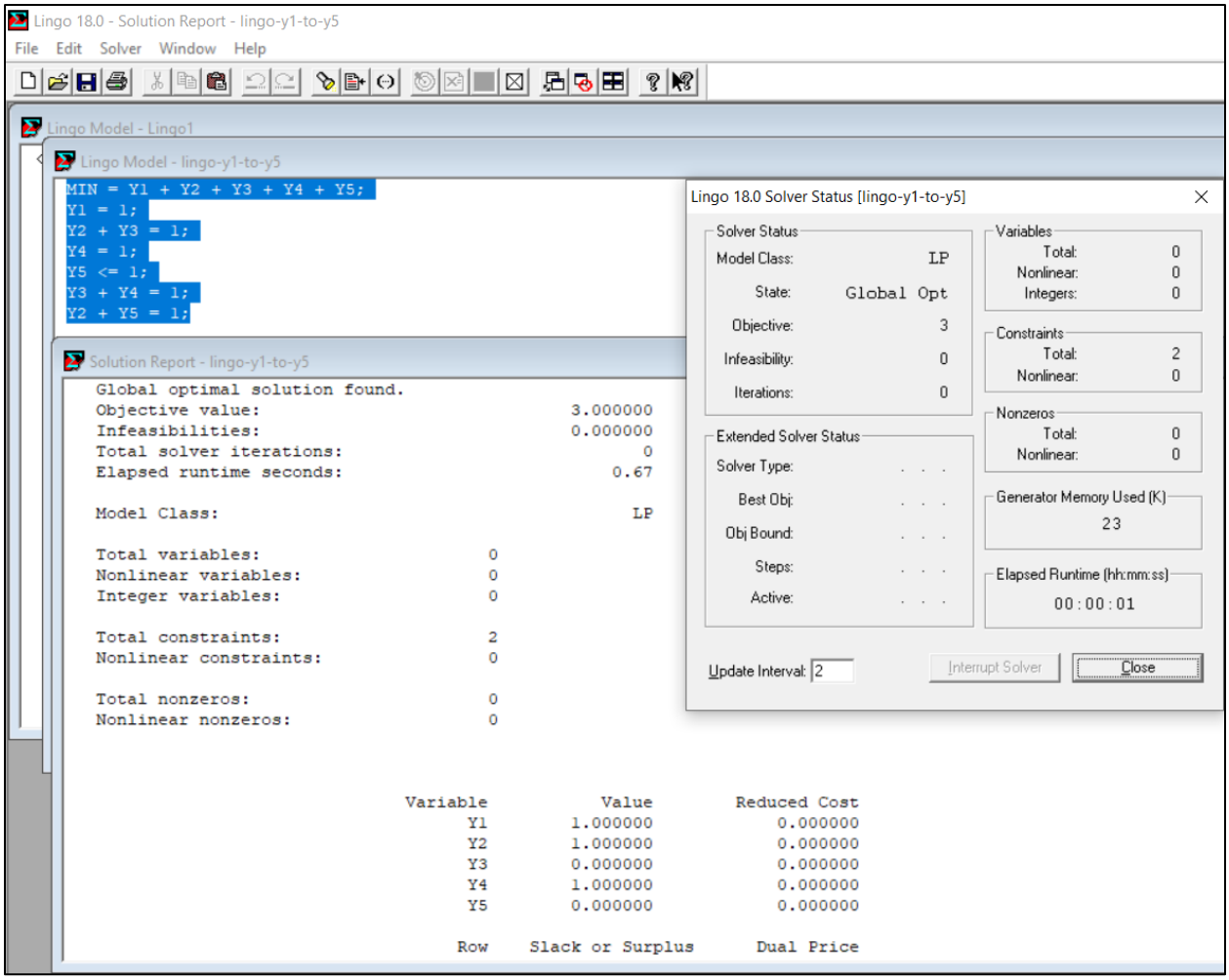
$y_5 \leq 1$

$y_3 + y_4 = 1$

$y_2 + y_5 = 1$ Result: $y_1 = 1$, $y_2 = 1$, $y_4 = 1$

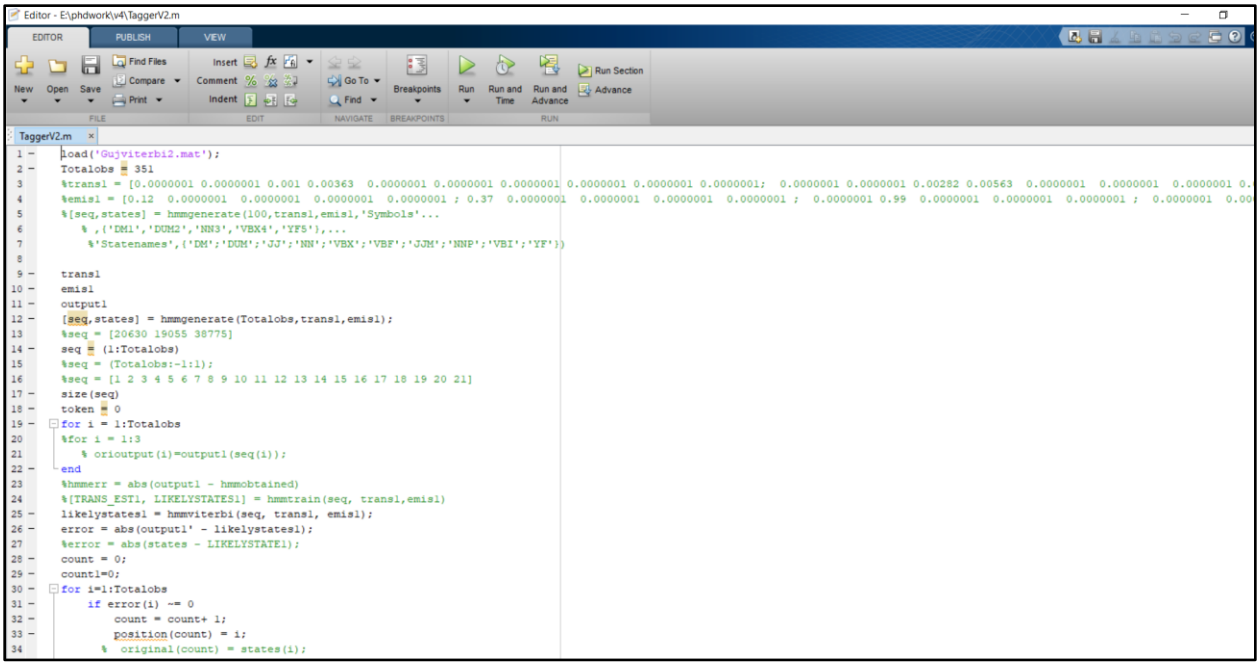
Same as Parsing tree show on Page No. 105

APPENDIX VIII : Sample Code of Lingo for Non-Projective Sentence 2



APPENDIX IX

Sample Code of MATLAB



References

[1] Abhilas Aswarth and M. Prashanth, “Bidirectional Dependency Parser for Indian Languages,” *Proc. ICON09 NLP Tools Contest Indian Lang. Depend. Parsing, Hyderabad, India, 2009.*, 2009.

[2] Aiyesha Sadiya, Anusha V Illur, Archana R Hegde, Ashwini R, and Chandini S B, “Survey on Natural Language Processing and its Applications,” *Int. J. Eng. Res.*, vol. V7, no. 01, pp. 379–389, 2018.

[3] B. R. Ambati, S. Husain, J. Nivre, and R. Sangal, “On the Role of Morphosyntactic Features in Hindi Dependency Parsing,” *First Work. Stat. Parsing Morphol. Rich Lang. (SPMRL 2010)*, no. June, pp. 94–102, 2010.

[4] V. K. Anil Krishna Eragani, “Improving malt dependency parser using a simple grammar-driven unlexicalised dependency parser,” *Proc. Int. Conf. Asian Lang. Process. 2014, IALP 2014*, pp. 211–214, 2014.

[5] G. Aniruddha, B. Pinaki, D. Amiava, and S. Bandyopadhyay, “Dependency Parser for Bengali : the JU System at ICON 2009,” *NLP Tool Contest ICON 2009*, 2009.

[6] Anish A, “Part of speech tagging for malayalam,” *AMRITA VISHWA VIDYAPEETHAM*, no. July, 2008.

[7] G.-M. Antonio, García-Orellana, C. J., González-Velasco, H. M., and Et.al., “Comparing feature extraction techniques and classifiers in the handwritten letters classification problem,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6354 LNCS, no. PART 3, pp. 106–109, 2010.

[8] A. Antonios and Zapranis Achilleas, “Wavelet neural networks: A practical guide,” *www.elsevier.com/locate/neunet Wavelet*, vol. 42, pp. 1–27, 2013.

[9] Archit Yajnik, “ANN Based POS Tagging For Nepali Text,” *Int. J. Nat. Lang. Comput.*, vol. 7, no. 3, pp. 13–18, 2018.

[10] Archit Yajnik, “Part of Speech Tagging Using Statistical Approach for Nepali Text,” *Int. J. Cogn. Lang. Sci.*, vol. 11, no. 1, pp. 76–79, 2017.

[11] ArchitYajnik, “General Regression Neural Network Based PoS Tagging for Nepali Text,” *CS IT-CSCP 2018*, pp. 35–40, 2018.

[12] ArchitYajnik, “Analysis of Verb Frame Lexicon for Nepali Shallow Parser Usinng Bipertite Graph”,*Internation Journal of Mind,Brain and Cognition*,Vol.9,NO.1-2, pp. 54-68, 2019.

[13] V. Ashwini and S. Husain, “A classification of dependency labels in the Hindi Treebank Hindi / Urdu

Treebank."

[14] D. B., *Introduction to graph theory*. 2002.

[15] B.MSagar and G Shobha, "Solving the Noun Phrase and Verb Phrase Agreement in Kannada Sentences," *Int. J. Comput. Theory Eng.*, vol. 1, no. 3, pp. 288–292, 2009.

[16] R. R. B.Venkata, "A hybrid dependency parsing approach for Telugu language," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 5, no. 4, pp. 77–85, 2015.

[17] BarbaraPlank, "Natural Language Processing : Introduction to Syntactic Parsing," *DISI,University of Trento*, 2012.

[18] BegumRafiya, P. Soma, and M. M. V. and M. B. A. for Viswanath Naidu, Mr. Gadde, "ICON09 NLP TOOLS CONTEST : INDIAN LANGUAGE DEPENDENCY PARSING Hyderabad , India," *Proc. ICON09 NLP Tools Contest Indian Lang. Depend. Parsing. Hyderabad, India. 2009.*, no. December, 2009.

[19] A. Bharati, S. Husain, B. Ambati, S. Jain, D. Sharma, and R. Sangal, "Two semantic features make all the difference in parsing accuracy," *Proc. 6th Int. Conf. Nat. Lang. Process.*, vol. 8, no. November, 2008.

[20] A. Bharati, M. Bhatia, V. Chaitanya, and R. Sangal, "Paninian Grammar Framework Applied to English," *South Asian Lang. Rev. Creat. Books, New Delhi, 1998.*, pp. 1–23, 1996.

[21] A. Bharati, M. Gupta, V. Yadav, K. Gali, and D. M. Sharma, "Simple parser for indian languages in a dependency framework," *ACL-IJCNLP 2009 - LAW 2009 3rd Linguist. Annot. Work. Proc.*, no. August, pp. 162–165, 2009.

[22] A. Bharati and R. Sangal, "Parsing free word order languages in the Paninian framework," *Dep. Comput. Sci. Eng.*, pp. 105–111, 1993.

[23] A. Bharti, *natural language processing*, vol. 66. 1994.

[24] P. Bhattacharyya, "Natural Language Processing: A Perspective from Computation in Presence of Ambiguity, Resource Constraint and Multilinguality," *CSI J. Comput.*, vol. 1, no. 2, pp. 1–11, 2012.

[25] R. Das Bishwa, S. Smrutirekha, P. Sekhar, and PatnaikSrikanta, "Part of speech tagging in odia using support vector machine," *Procedia Comput. Sci.*, vol. 48, no. C, pp. 507–512, 2015.

[26] J. D. Bodapati and N. Veeranjaneyulu, "Feature extraction and classification using Deep convolutional Neural Networks," *J. Cyber Secur. Mobil.*, vol. 8, no. 2, pp. 261–276, 2019.

[27] G. R. Chaudhari Shaileshrds Extraction from printed Bilingual Gujarati-Roman Script Documents," *VNSGU J. Sci. Technol.*, vol. 4, no. 1, pp. 83–94, 2015.

[28] ChenWenliang, L. Zhenghua, and M. Zhang, “Dependency Parsing : Past , Present , and Future,” *Soochow Univ.*, 2009.

[29] N. Choudhary, “Human Language Technology Challenges for Computer Science and Linguistics - 5th Language and Technology Conference, LTC 2011, Revised Selected Papers,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8387 LNAI, no. February 2015, 2014.

[30] M. Croitoru and E. Compatangelo, “A tree decomposition algorithm for Conceptual Graph projection,” *Proceedings, 10th Int. Conf. Princ. Knowl. Represent. Reason. KR 2006*, pp. 271–276, 2006.

[31] M. Croitoru and K. van Deemter, “A Conceptual Graph Approach to the Generation of Referring Expressions,” *IJCAI-07*, pp. 2456–2461, 2007.

[32] CroitoruMadalina *et al.*, “A Conceptual Graph Based Approach to Ontology Similarity Measure,” *ICCS 2007, LNAI 4604*, vol. 4604, no. di, pp. 154–164, 2007.

[33] CroitoruMadalina and C. Ernesto, “Conceptual Graph Assemblies digs,” *Univ. Aberdeen King’s Coll. Aberdeen, AB243UE Abstr.*, 2002.

[34] K. Dhanashree, “Specifying Context free Grammar for Marathi Sentences,” *Int. J. Comput. Appl. (0975 – 8887)*, vol. 99, no. 14, pp. 38–41, 2014.

[35] Dholakia Jignesh, Yajnik Archit, and Negi Atul, “Wavelet Feature Based Confusion Character Sets for Guja[1] J. Dholakia, ‘Wavelet Feature Based Confusion Character Sets for Gujarati Script,’ pp. 371–376, 2007.rati Script,” *Int. Conf. Comput. Intell. Multimed. Appl. 2007 Wavelet*, pp. 371–376, 2007.

[36] R. Dieng and S. Hug, “Comparison of «Personal Ontologies» Represented through Conceptual Graphs,” *Eur. Conf. Artif. Intell.*, pp. 341–345, 1998.

[37] EkbalAsif and Bandyopadhyay Sivaji, “Part of speech tagging in Bengali using Support vector Machine,” *Proc. - 11th Int. Conf. Inf. Technol. ICIT 2008*, pp. 106–111, 2008.

[38] H. Eristi, U. Aysegul, and DemirYakup, “Wavelet-based feature extraction and selection for classification of power system disturbances using support vector machines,” *Electr. Power Syst. Res. www.elsevier.com/locate/epsr Wavelet-based*, vol. 80, no. 7, pp. 743–752, 2010.

[39] G. Fănică, “Generating the maximum spanning trees of a weighted graph,” *J. Algorithms*, vol. 8, no. 4, pp. 592–597, 1987.

[40] P. Gadde, K. Jindal, SamarHusain, M. Dipti, and SangalRajeev, “Improving Data Driven Dependency Parsing using Clausal Information,” *Hum. Lang. Technol. 2010 Annu. Conf. North Am. Chapter Assoc.*

Comput. Linguist., no. June, pp. 657–660, 2010.

[41] Hall Keith B, "K-best spanning tree parsing," *Proc. 45th Annu. Meet. Assoc. Comput. Linguist.*, vol. 45, no. May, pp. 392–399, 2007.

[42] F. M. Hasan, N. UzZaman, and M. Khan, "Comparison of Unigram, Bigram, HMM and Brill's POS Tagging Approaches for some South Asian Languages," *Conf. Lang. Technol.*, no. June, 2007.

[43] Haykin Simon, *Neural Networks and Learning Machines*, vol. 3. 2008.

[44] A. P. J, N. J. Warrier Assistant Professor, and S. K. P Professor, "Penn Treebank-Based Syntactic Parsers for South Dravidian Languages using a Machine Learning Approach," *Int. J. Comput. Appl.*, vol. 7, no. 8, pp. 975–8887, 2010.

[45] D. Kapur, *Lecture Notes in Artificial Intelligence: Preface*, vol. 5081 LNAI. 2008.

[46] D. A. Karras, "Improved Defect Detection Using Support Vector Machines and Wavelet Feature Extraction Based on Vector Quantization and SVD Techniques," *Proc. Int. Jt. Conf. Neural Networks*, vol. 3, pp. 2322–2327, 2003.

[47] P. Kosaraju, R. K. Sruthilaya, V. Bhargav, and A. Reddy, "Experiments on Indian Language Dependency Parsing," *Lang. Technol. Res. Centre, IIIT-Hyderabad, India.*, 1995.

[48] B. Krishna, "Structure of Nepali grammar," *Work. Pap. 2004-2007*, pp. 332–396.

[49] Madalina Croitoru, "Conceptual Graphs at Work : Efficient Reasoning and Applications," *Dep. Comput. Sci.*, 2006.

[50] A. Mahajan, Sharmistha, and S. Roy, "Feature Selection for Short Text Classification using Wavelet Packet Transform," *Proc. ofthe 19th Conf. Comput. Lang. Learn. pages 321–326, Beijing, China, July 30-31, 2015*, pp. 321–326, 2015.

[51] Mannem Prashanth, H. Chaudhry, and BharatiAkshar, "Insights into non-projectivity in Hindi," *ACL-IJCNLP 2009 - Jt. Conf. 47th Annu. Meet. Assoc. Comput. Linguist. 4th Int. Jt. Conf. Nat. Lang. Process. AFNLP, Proc. Conf.*, no. August, pp. 10–17, 2009.

[52] Mcdonald Graham and Macdonald Craig, "A Study of SVM Kernel Functions for Sensitivity Classification Ensembles with POS Sequences," *SIGIR '17, August 07-11, 2017, Shinjuku, Tokyo, Japan*, no. June, pp. 7–11, 2017.

[53] V. Meher and DeepakKalyan, "Constraint based Hindi dependency parsing," *Lang. Technol. Res. Cent.*, pp. 1–6, 2008.

[54] A. Mukherjee, S. Kübler, and M. Scheutz, “POS Tagging Experts via Topic Modeling,” *Proc. 13th Int. Conf. Nat. Lang. Process.*, pp. 120–128, 2016.

[55] A. Nietzio, “Support Vector Machines for Part-of-Speech Tagging,” *Ruhr-Universitat bochum*, pp. 5–8, 2002.

[56] J. Nivre, “Parsing Indian Languages with MaltParser,” *Proc. ICON09 NLP Tools Contest Indian Lang. Depend. Parsing*, no. January, pp. 12–18, 2009.

[57] J. Nivre, J. Hall, and N. Jens, “MaltParser : A Data-Driven Parser-Generator for Dependency Parsing,” *Sch. ofMathematics Syst. Eng.*, pp. 2216–2219.

[58] J. Nivre *et al.*, *MaltParser: A language-independent system for data-driven dependency parsing*, vol. 13, no. 2. 2007.

[59] Nongmeikapam Kishorjit and BandyopadhyaySivaji, “Manipuri Chunking : An Incremental Model with POS and RMWE,” *https://www.researchgate.net/publication/272793524*, no. December, pp. 277–286, 2014.

[60] K. Nongmeikapam and S. Bandyopadhyay, “Genetic Algorithm (GA) Implementation for Feature Selection in Manipuri POS Tagging,” *Proc. 13th Int. Conf. Nat. Lang. Process.*, no. December, pp. 267–274, 2016.

[61] A. B. P. R. Ray V. Harish and S. Sarkar, “Part of Speech Tagging and Local Word Grouping Techniques for Natural Language Parsing in Hindi,” *Proc. ICON 2003*.

[62] P. J. Antony and K. P. Soman, ‘Kernel based part of speech tagger for Kannada,’ 2010 Int. Conf. Mach. Learn. Cybern. ICMLC 2010, vol. 4, no. July, pp. 2139–2144, 2010

[63] V. Pandey, M. V. Padmavati, and R. Kumar, “Rule based parts of speech tagger for Chhattisgarhi language,” *Int. J. Recent Technol. Eng.*, vol. 7, no. 4, pp. 192–194, 2018.

[64] PatelChirag and G. Karthik, “Part-Of-Speech Tagging for {G}ujarati Using Conditional Random Fields,” *Proc. IJCNLP-08 Work. NLP Less Privil. Lang.*, no. January, pp. 117–122, 2008.

[65] S. Petrov, “Syntax and Parsing,” *Comput. Linguist.*, no. 1979, pp. 577–578, 1995.

[66] A. PJ and S. KP, “Computational Morphology and Natural Language Parsing for Indian Languages : A Literature Survey,” *Int. J. Sci. Eng. Res.*, vol. 3, no. 3, pp. 136–146, 2012.

[67] K. Prakash, D Akila, and P. Rajesh, “Advances in natural language processing –a survey of current research trends, development tools and industry applications,” *Int. J. Recent Technol. Eng.*, vol. 7, no. 5C, pp. 199–201, 2019.

[68] Purva S.Dholakia and MohamedYoonus, “Rule Based Approach for the Transition of Tagsets to Build the

{POS} Annotated Corpus," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 3, no. 7, pp. 7417–7422, 2014.

[69] B. Rafiya and S. D. Misra, "Development and Analysis of Verb Frame Lexicon for Hindi," *Linguist. Lit. Stud. 5(1) 1-22, 2017*, vol. 5, no. 1, pp. 1–22, 2017.

[70] B. Rafiya, H. Samar, B. Lakshmi, and S. D. Misra, "Developing Verb Frames for Hindi," *Lang. Technol. Res. Centre,* pp. 1925–1932, 2003.

[71] Raghu Pujitha Gade, "Dependency parsing approaches for Indian Languages : Hindi and Sanskrit," *Int. Inst. Inf. Technol. Hyderabad - 500 032, INDIA July*, no. July, 2014.

[72] M. Rahman, S. Das, and U. Sharma, "Parsing of part-of-speech tagged Assamese Texts," *J. Comput. Sci.*, vol. 6, no. 1, 2009.

[73] K. M. Rajpoot and N. M. Rajpoot, "Wavelets and support vector machines for texture classification," *Proc. INMIC 2004 - 8th Int. Multitopic Conf.*, pp. 328–333, 2004.

[74] A. B. Ram, HusainSamar, S. Jain, S. D. Misra, and SangalRajeev, "Two methods to incorporate local morphosyntactic features in Hindi dependency parsing," *Proc. NAACL HLT 2010 First Work. Stat. Parsing Morphol. Lang. (SPMRL 2010)*, no. c, pp. 22–30, 2010.

[75] Rishikesh, "Parts Of Speech Tagger for Maithili Language Using HMM," *Int. J. Innov. Adv. Comput. Sci. IJIACS*, vol. 7, no. 4, pp. 206–211, 2018.

[76] D. Romero, N. Galeano, and A. Molina, "Wavelet Neural Networks and their application in the study of dynamical systems David," *Methods Tools Collab. Networked Organ.*, no. August, pp. 69–90, 2008.

[77] B. M. Sagar, G. Shobha, and R. Kumar, "Context Free Grammar (CFG) Analysis for simple Kannada sentences," *Proc. Int. Conf. [ACCTA-2010] Spec. Issue IJCCT*, vol. 1, no. 2, p. 2, 2010.

[78] SahariaNavanath, SharmaUtpal, and J. Kalita, "A First Step Towards Parsing of Assamese Text," *Dep. Comput. Sci. Eng. Tezpur Univ.*

[79] J. Sambhav *et al.*, "Exploring Semantic Information in Hindi WordNet for Hindi Dependency Parsing," *Lang. Technol. Res. Cent. IIIT Hyderabad*, 2013.

[80] K. Samina, KhalilbTehmina, and N. Shamila, "A survey of feature selection and feature extraction techniques in machine learning," *Proc. 2014 Sci. Inf. Conf. SAI 2014*, pp. 372–378, 2014.

[81] SanjayChatterji, RoyDevshri, PraveenSonare, and SudeshnaSarkar, "Grammar Driven Rules for Hybrid Bengali Dependency Parsing," *Proc. ICON-2009 7th Int. Conf. Nat. Lang. Process. Macmillan Publ. India.*, 2009.

[82] D. Sankare, DharArnab, and GarainUtpal, “Structure Simplification and Demand Satisfaction Approach to Dependency Parsing in Bangla,” 2008.

[83] C. Schleicher, “An introduction to wavelets for economists,” *Bank canada*, no. Working paper, p. 32, 2002.

[84] A. Secer, A. C. Sonmez, and H. Aydin, “Ontology mapping using bipartite graph,” *Int. J. Phys. Sci.*, vol. 6, no. 17, pp. 4224–4244, 2011.

[85] M. SELVAM, A. M. NATARAJAN, and R. THANGARAJAN, “Structural Parsing of Natural Language Text in Tamil Language Using Dependency Model,” *Int. J. Comput. Process. Lang.*, vol. 22, no. 02n03, pp. 237–256, 2009.

[86] B. ShahiTej, D. Nath, and BalamiBikash, “Support Vector Machines based Part of Speech Tagging for Nepali Text,” *Int. J. Comput. Appl. (0975 – 8887)*, vol. 70, no. 24, pp. 38–42, 2013.

[87] N. Shimizu, “Maximum Spanning Tree Algorithm for Non-projective Labeled Dependency Parsing,” *Comput. Linguist.*, no. June, pp. 236–240, 2006.

[88] Singla Karan, Tammewar Aniruddha, Jain Naman, and S. Jain, “Two-stage Approach for Hindi Dependency Parsing Using MaltParser,” *https://www.researchgate.net/publication/235987545 Two-stage*, no. December, pp. 163–170, 2012.

[89] S. W. Smoliar, “Conceptual structures: Information processing in mind and machine,” *Artif. Intell.*, vol. 33, no. 2, pp. 259–266, 1987.

[90] J. Sowa, “Basic Conceptual Graphs.”

[91] J. F. Sowa, “Conceptual Graphs for a Data Base Interface.,” *IBM J. Res. Dev.*, vol. 20, no. 4, pp. 336–357, 1976.

[92] J. F. Sowa, “Conceptual Graphs,” *Found. Artif. Intell.*, vol. 3, no. Findler 1979, p. 213−237, 2008.

[93] J. F. Sowa and E. C. Way, “Implementing a Semantic Interpreter Using Conceptual Graphs.,” *IBM J. Res. Dev.*, vol. 30, no. 1, pp. 57–69, 1986.

[94] S. B. Swapnali Deelip, “Comparative Study of Rule Based Approach for Grammar Checker,” *Int. J. Manag. Technol. Eng.*, vol. IX, no. I, pp. 1315–1319, 2019.

[95] R. Swati, R. Komal, and D. Rajesh, “Lexicon Parser for syntactic and semantic analysis of Devanagari sentence using Hindi wordnet,” *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 3, no. 4, 2014.

[96] S. Tembhekar and M. Kanojiya, “A Survey Paper on Approaches of Natural Language Processing (NLP),” *Int. J. Adv. Res. Ideas Innov. Technol. ISSN*, vol. 3, no. 3, pp. 1496–1498, 2017.

[97] Valeeva Marina, “Graph-Based Dependency Parsing.”

[98] Vasunthira Devi and Ponnusamy Ramalingam, “EHMM : An Analysis the Tamil Tweets using Enhanced Hidden Markov Model,” *https//www.researchgate.net/publication/338711652 EHMM*, vol. 119, no. 18, pp. 1331–1338, 2018.

[99] B. Venkata, S. Kumari, and R. R. Rao, “Improving Indian Language Dependency Parsing by Combining Transition-based and Graph-based Parsers,” *Int. J. Comput. Appl.*, vol. 115, no. 5, pp. 975–8887, 2015.

[100] S. K. B. Venkata and R. R. Ramisetty, “Hindi Dependency Parsing using a combined model of Malt and MST,” *Proc. Work. Mach. Transl. Parsing Indian Lang. (MTPIL-2012), pages 171–178, COLING*, no. December, pp. 171–178, 2012.

[101] G. Vipul, J. Rutva, and E. Patel, “A Review on Part-Of-Speech Tagging on Gujarati Language,” *Int. Res. J. Eng. Technol.*, vol. 6, no. 6, 2019.

[102] Wael Etaiwi and A. Arafat, “Graph-based Arabic NLP Techniques: A Survey,” *Procedia Comput. Sci.*, vol. 142, pp. 328–333, 2018.

[103] S. Y.Hala and S. H.Virani, “International Journal of Advance Engineering and Research Development,” *Int. J. Adv. Eng. Res. Dev.*, vol. 2, no. 5, pp. 464–467, 2015.

[104] T. Yasuyuki, Y. Ryosuke, KitamuraTakuya, and A. Higeo, “feature extration usimg syupport vector Machines,” *ICONIP 2010, Part II, LNCS 6444, pp. 108–115, 2010.*, pp. 108–109, 2010.

List of Publications

[1] **M. prajapati** Archit Yajnik, "Part of Speech Tagging Using Statistical Approach for Gujrati Text," International Journal Of Applied Research In Science And Engineering ISSN (Online): 2456-124X vol. 11, no. 1, pp. 76–79, 2017.

[2] Y. Archit, P. Ashish, and **P. Manisha**, "A Conceptual Graph Approach to the Parsing of Projective Sentences," International Journal of Mathematics and Computer Science, 15(2020), no. 1, – (**IJMCS**) A vol. 15, no. 1, 2020.

[3] A. Y. **Manisha Prajapati**, "POS Tagging of Gujarati Text using VITERBI and SVM," International Journal of Computer Applications (0975 – 8887) Volume 181 – No. 43, March 2019

[4] **Manisha prajapati** and Archit Yajnik, "Parsing for Indian Languages A Literature Survey," International Journal of Computer Sciences and Engineering(IJCSE)., vol. 6,issue no- 8, pp. 1009–1018, Aug 2018.

[5] **Manisha prajapati** and Archit Yajnik, "Constraint-Based Gujarati parser Using LPP," Proceedings of First International Conference on Computing, Communication and Cyber-Security, **Springer** (IC4S 2019), 978-981-15-3368-6, 487124_1_En (29)