Sanskrit Language Enconversion to Universal Networking Language (UNL)

A Thesis

submitted in partial fulfillment of the requirements for the award of degree of

Doctor of Philosophy

by

SITENDER

(Regn. No. : 950903033)

under the guidance of

Dr. Seema Bawa

Professor, Computer Science and Engineering Department,

Thapar Institute of Engineering and Technology, Patiala-147004, INDIA

Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, Patiala -147004, INDIA

October 2021

Certificate

I hereby certify that the work which is being presented in this thesis entitled "**Sanskrit Language Enconversion to Universal Networking Language (UNL)**", in partial fulfillment of the requirement for the award of degree of "**Doctor of Philosophy**" submitted in Computer Science and Engineering Department, Thapar Institute of Engineering and Technology, Patiala, India, is an authentic record of my own work carried out under the supervision of **Dr. Seema Bawa** and refers other research works which are duly listed in the reference section. The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(Sitender)

Regn. No. 950903033

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

25/10/2024

(Dr. Seema Bawa)

Supervisor

Professor, Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, Patiala, 147004, Punjab, INDIA.

Acknowledgement

It gives me an immense pleasure while acknowledging the contribution of persons those helped me to achieve my research goal.

First of all, I would like to thank my supervisor Dr. Seema Bawa, Professor Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala, for her patience, motivation and immense knowledge. Without her support and motivation it was never possible for me to pursue research goals. She always guided me as a parent in every phase of my research work.

I would like to thank Dr. Maninder Singh, Head of Department, Computer Science and Engineering, Thapar Institute of Engineering and Technology, PAtiala, for providing me the healthy research environment and academic facilities at TIET campus. Without his support and motivation it was never possible for me to pursue research goals.I gratefully pay my sincere thanks to the members of Doctoral Committee members for their observations, constructive criticism and valuable comments which helped me a lot in presenting the results and shaping this thesis.

My sincere thanks also goes to Dr. Sushma Jain, PhD Coordinator, CSED, TIET and Dr. Ravinder Kumar, Associate Professor, CSED, TIET, who provided me an opportunity to pursue my Ph.D. I would like to thank Mr. Vijay Soni, Mr. Sandeep Sahran, Dr. Ajay Kumar, Mr. Sachender and all my dear colleagues pursuing Ph.D. at TIET. Their constant support always helped me to move on. I would like to thank my family for their support. Without their support this day of submitting my thesis would have never been possible.

(Sitender)

Contents

List of Tables

List of Figures

Acronyms

AI Artificial Intelligence. 1

ANN Artificial Neural Network. 74

BiLSTM Bidirectional Long Short Term Memory. ix, 90, 150, 151

BLEU Bilingual Evaluation Understudy. ii, vii, 29, 49, 50, 52, 53, 67–70, 72, 75, 76, 79, 148, 150, 151, 154, 155, 157, 159

CBMT Corpus Based Machine Translation. v, 38, 63, 157

CFG Context Free Grammar. 60, 92

CNF Chomsky Normal Form. 92

CYK Cocke-Younger-Kasami. ii, v, 30, 91–93, 146

DMT Direct Machine Translation. v, 11, 35, 37, 41, 42, 147, 157

EBMT Example Based Machine Translation. 39, 40, 63–65, 69, 82

HBMT Hybrid approach Based Machine Translation. v, 71

IBMT Interlingua Based Machine Translation. 37, 44

LSTM Long Short Term Memory. viii, 40, 77, 88, 89

MT Machine Translation. i, ii, iv, v, vii, viii, 1, 2, 4, 5, 7, 10, 11, 29–32, 34, 35

MTS Machine Translation System. iv, v, vii, 1, 7, 10, 30, 31, 40–46, 48, 50, 52, 55–57, 62, 64, 66, 67, 69, 73–76, 78–81, 83, 87, 148, 149, 159

NL Natural Language. i

NLP Natural Language Processing. 1, 77, 79

NMT Neural Machine Translation. v, 40, 76, 77, 81

POS Part-of-Speech. ii, 29, 30, 51, 59, 60

RBMT Rule Based Machine Translation. v, 11, 36, 44, 54–58, 60, 61, 69–72, 74, 81, 157

RNN Recurrent Neural Network. 40

SER Sentence Error Rate. 66

SLSTM Stacked Long Short Term Memory. ix, 90, 150, 151

SMT Statistical Machine Translation. 11, 38, 63, 65–72, 76, 81

SOV Subject Object Verb. 4, 9, 55, 75, 143, 144

SVO Subject Verb Object. 9, 55, 75

TBMT Transfer Based Machine Translation. 36, 37, 44, 53

UNL Universal Networking Language. i, ii, iv–ix, 1, 9–15, 22–30, 46–53, 80, 82–97, 100, 101, 105–107, 113, 114, 126–128, 132, 133, 145–147, 151, 154, 155, 158–160

UNLKB UNL Knowledge Base. 12, 13

UNU United Nations University. i

UW Universal Word. vi, vii, 12–20, 23, 25–29, 46, 49–51, 80, 95, 96, 98, 99, 105, 106, 108, 122, 124, 125, 127, 131, 132

WER Word Error Rate. 66

Abstract

Machine Translation (MT) has been the prime research area in last few decades. Researchers from different domains like statistics, linguistics, mathematics, artificial intelligence and philosophy have been witnessed to work on solving various problems related to MT. Several methodologies have been used by researchers to develop MT systems for different languages. Developing MT system based on Universal Networking Language (UNL) is also an effort in the direction of MT field. UNL was first launched in 1996 by United Nations University (UNU) at Institute of Advanced Studies, Tokyo Japan. Key components of UNL for natural language processing are EnConverter and DeConverter. The first component is used to convert the Natural Language (NL) sentence into equivalent UNL statements and the second component performs the reverse operation i.e. generates the NL from UNL expressions. The focus of the research work carried out in this thesis is on the development of Enconverter system for Sanskrit language.

The thesis starts with introduction part which provides information about the importance of machine translation in today's multilingual world, Sanskrit language structure, UNL system, need of MT, problems faced during MT development and the comparison of UNL with other systems. This work also highlights a comprehensive survey of MT approaches, existing MT systems, linguistic tools, data repositories and MT platforms. Among the available research in machine translation system, it is found that a little work has been done by the researchers for Sanskrit language MT development. The work that has been done, does not take care of application of neural network for designing stemmer, tagger, parser as well as translator for developing Sanskrit MT system. Further keeping in mind the research gaps from the survey there is a need to develop a new Sanskrit MT system which could perform translation in multiple languages simultaneously with less effort. In this work, a new MT system "SANSUNL" for Sanskrit language is proposed which translates Sanskrit to UNL

expressions. The proposed "SANSUNL" MT system consists of seven layers. Each layer is having its unique functionality that includes pre-processing, Part-of-Speech (POS) tagging, parsing, node-list creation, case marker identification, unmatched word handling and UNL expression generation. It uses state-of-the-art technology i.e. neural network for POS tagging and CYK parsing algorithm for language parsing. A new Sanskrit grammar and a new algorithm is proposed for parsing and generating the parse tree for Sanskrit text. The system also uses a new stemmer to find the base form of words to perform the shallow parsing of input text. To test and evaluate the proposed system five data-sets has been used. The system is evaluated using both traditional methods which includes Fluency score and adequacy score as well as automatic evaluation method Bilingual Evaluation Understudy (BLEU) score. From result analysis it is found that the proposed system is performing very well and effectively translate Sanskrit text to UNL expressions. A new Sanskrit to English MT system is also proposed which uses layers of "SANSUNL" system. The system is tested on 500 Sanskrit sentence data-set and evaluated using fluency, adequacy and BLEU score. Finally the thesis is concluded with the perspective of future work.

Chapter 1

Introduction

This chapter familiarises with the role of machine translation in today's world. It also includes an overview of Sanskrit language, Sanskrit EnConversion, Universal Networking Language (UNL), need of Machine Translation (MT) and problems faced in developing Machine Translation System (MTS). Chapter-wise organization of the thesis and research contributions are highlighted at the end of this chapter.

1.1 Overview of the study

With the evolution of computer technology in 21st century, dependency of human on computer is increasing day by day. The availability of computer systems with high computational power has opened various new research areas and encouraging people all over the world to provide efficient solutions to various problems related to social, cultural, economic, defense, education and communication. Natural languages have shown a vital role in shaping human social behavior as they prepare the necessary mechanism for day to day communication among human beings [8]. Natural Language Processing (NLP) comprises of three basic components : processing, understanding and generation [9]. NLP is a sub-domain of Artificial Intelligence (AI) and Machine Translation (MT) is one of the application of NLP. MT uses computer systems fully or partially to provide automated or semi-automated translations among various natural languages. MT is also one of the predominant and very challenging filed of research which has fascinated researchers all over the world to work in

this domain. MT has seen different phases of development with focus on different approaches. According to Vauquois triangle; MT approaches have different level of complexities and the complexity increases from bottom to top [10]. Direct MT approach is at bottom, Transfer based MT is in middle while the top position is occupied by Interlingua Based MT approach of Vauquois triangle. The work done in this thesis demonstrates an Enconverter System "SANSUNL" for the Sanskrit Language. Sanskrit is one of the oldest and morphological rich language in world. Devanagari script has been used for Sanskrit language.

1.2 Sanskrit: The "Samaskrita"

Sanskrit word is a combination of "Sam" which means perfection or entirely and "krit" which means done, so Sanskrit means completely done or entirely done. Sanskrit language was assumed to be Dev Vani due to the assumption that loard Brahma had introduced it in the world. The grammar of Sanskrit is Panini grammar developed by great sage Panini. Sanskrit has 36 consonants and 16 vowels which have never changed or updated since their inception. Morphology of Sanskrit language is so strong and unique that infinite number of words could be generated from root words called Pada and Dhatu simply by adding prefixes and suffixes. There are more than twenty two hundred root verbs with 90 forms of each that is used to describe any action in the world and 21 forms of each noun or pronoun to generate any word. The word in Sanskrit corresponds to the properties of an object not the object itself [11]. Sanskrit language had produced large amount of manuscripts even before the invention of printing press and the content was one hundred times larger than the combined content of Greek and Latin language. Sanskrit text includes poetry, mathematical, philosophical, scientific, political, medical research and meditation information in the form of Vedas, Puraans and Upnishads. Sanskrit is one of the officially recognized language in India and according to census 1991 of India approximately fifty thousand persons have been identified who were using Sanskrit as primary language for communication purpose [12]. Sanskrit is rich in literary criticism; Bharata's Natya Shastra is one of the oldest Indian

critic in 5th century AD [13]. Sanskrit language has been accepted as a most powerful and scientific language of the world. Due to enhancement in digital technology, Sanskrit text is now available online and the researchers are taking full utilization of this online Sanskrit text to develop several linguistic processing tools. There is a strong need of translating this online text into other languages to develop new techniques in medical science, mathematics, political science etc. by using the hidden treasure of Sanskrit [14].

In Sanskrit, words are divided into three main categories as shown in Figure 1.1.

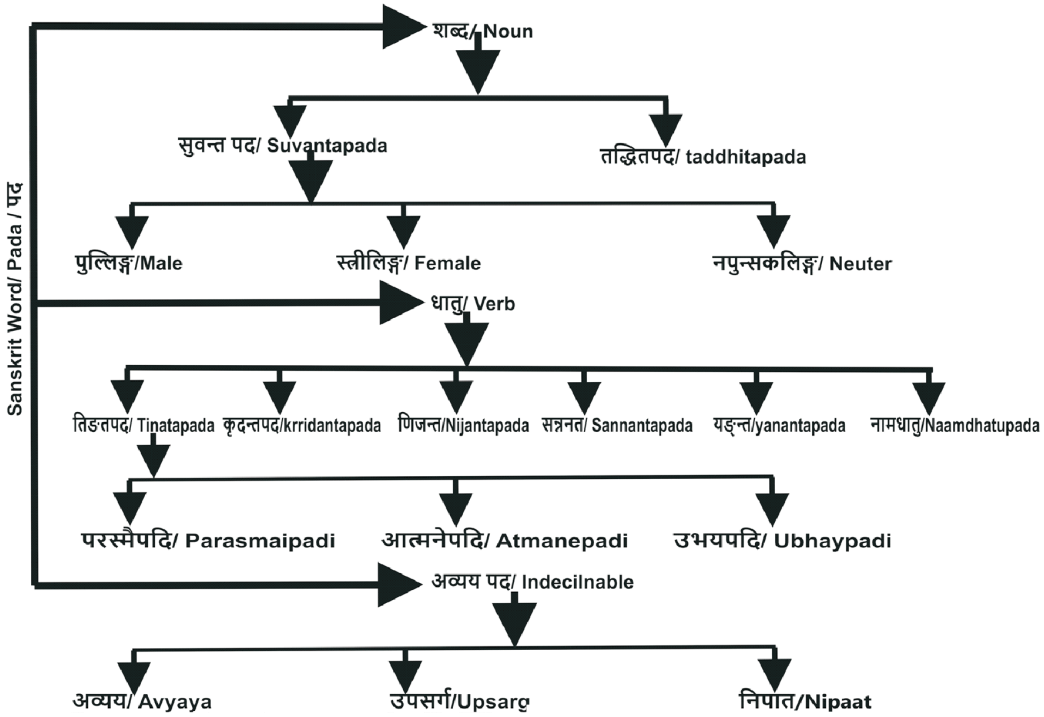

Figure 1.1: Sanskrit Word Architecture

The first category of word is noun. In Sanskrit, there are eight cases known as Vibhakti's which are used to find out the case of any noun. For each noun, there are three numbers as singular, dual and plural which counts $8*3=24$ forms of the noun. There are three genders as well: masculine, feminine and neuter which again gives $3*24=72$ forms of noun in Sanskrit. Nouns are further categorized as suvant pada and tadhipada. Former has suffix sup and the later has a different suffix attached to the root to make base form of the word. The noun are also classified based on their endings such as ending with अ (a) are known as अकारान्त

(akaaraantaH), ending with इ (i) is called इकारान्तः (ikaaraantaH) and so on.

The second category of word is verb. The root verb in Sanskrit is known as Dhatu (धातु) and generating verb from the root is known as Dhatroop (धातुरुप). There are six tenses and four moods in the Sanskrit language. There are ten classes of verbs in Sanskrit as Bhavaadi, Gunaadi, etc. The verbs can be classified as transitive/ सकर्मक or intransitive verb / अकर्मकधातु. Verbs in Sanskrit has no gender. Sanskrit verbs have three numbers as singular, dual and plural and three persons as first, second and third. The verb forms generated by adding tip (तिप्) suffix are known as तिङन्त(tinant). Based on recipient of verb's action, verbs are further categorized as Parasmaipadi or Aatmanepadi or Ubhapadi, if the recipient is subject itself, or object or both respectively. Adjective in Sanskrit are special case of nouns. There is no category for adjectives in Sanskrit language. The word order in Sanskrit is generally SOV, although the language is free word order.

The third category of word in Sanskrit is known as अव्यय / Indeclinable words. अव्यय are the words in Sanskrit which does not change their form with gender, number, case etc. अव्यय are further categorized into three groups as: अव्यय (independent indeclinable), उपसर्ग (Prefix) and निपात (secondary suffix).

1.2.1 Need of Machine Translation (MT)

In a multilingual country like India where 22 languages have been used officially in order to remove the language barrier one of the solution is MT. This section highlights some of the applications of MT.

(i) **Machine Translation in Defense**

To stop the terrorist activities all over the world, security forces need to be more intelligent and technologically advanced. Security forces have to work in different regions of the world and terrorists uses their native languages to operate terror activities. Due to language differences, it becomes very difficult for the security forces to understand their discussion and take necessary action in stipulated time. So there is

a huge demand of MT systems in defence sector which could help the security forces to take necessary action on time to save human life. US government took the initiative to fund private organizations for developing new technologies which will help the security forces to protect their nation [15]. Similarly Indian government has also started funding different organizations to produce efficient machine translation systems for the security forces [16, 17].

(ii) **Machine Translation a need of commercial sector**

With the evolution of internet, industries have also started moving towards globalization from localization but the language barrier has been a big hurdle in the growth of the business. Industries have to invest a big amount on the translation of their product manuals and also to communicate with persons from different language backgrounds and this process was very time consuming as well. So there is a great need of machine translation systems which will perform the task of translation quickly, efficiently and automatically or with small human interventions. History shows great increase in revenue of the companies who has adopted machine translation systems on time and now they become world leader. Adobe is one of the example which shows 50% increase in speed of producing product documentation. It simply uses machine translation with post editing and resulted in increased revenue [18]. So machine translation plays a key role in commercialization of products in a multi-lingual environment all over the world [17].

(iii) **To remove the language barrier in a multi-lingual environment**

Facebook, Twitter, Instagram, Whatsapp, LinkedIn etc. are becoming common place for the people of different regions with different language backgrounds. These applications are used to share their views on common topics but the language is the major problem in such environment. To remove this language barrier, there is a need of machine translation system which the users can use for communication purpose [17].

(iv) **Machine Translation as an ancient culture and knowledge preserver**

In regions like India, where multiple languages are being used for communication purpose, dominance of one or two languages over others demoralizes the persons speaking less dominant languages. For example, dominance of Hindi and English language over Sanskrit language demoralises Sanskrit speakers and after some time the number of speakers of Sanskrit language reduces and later on leads to non-existence of such languages in the society. This loss is not only the language loss but also the culture associated with that language. So machine translation becomes the necessity in such cases to preserve such ancient languages and culture where a particular group of population wishes to levy a specific language over other members [17].

(v) **Machine Translation as an assistant to human translators**

Humans are having the limitation of memory. For a single human being it is very difficult to be perfect in multiple languages. Hiring individual human translator for each language in a multilingual environment is very costly and time consuming. Even the coordination among multiple human translators is again a big challenge. So to come out from such situations the machine translation systems can help the human translators as an assistant with no memory limitation and computation capability [19].

(vi) **Machine Translation as a nation builder**

For the development of any nation, implementation of all the policies and availability of resources should reach to every corner of nation. There should not be any communication gaps among states and the center. But in a multilingual environment, like in India, communication gap among people due to language barrier puts break on the development of the nation. The information about the latest technologies is not able to reach to its users like to farmers from the scientists. This is the main reason the Indian government decided to fund different research organizations for the development of machine translation system to bridge the gap of communication due to multiple languages [19].

(vii) **Machine Translation in health sector**

The availability of proficient medical translators at every place is impossible. So for medical doctors, language becomes a barrier in the treatment of patients and they even avoid using any machine translation system in communication with patients. But machine translation systems can help them to study the history and previous treatments if any, by asking the patient to enter the answers of some predefined question set in a machine translation system. Machine translation can also help chemists to understand doctor's prescriptions and the labels on the medicines [20].

(viii) **Machine Translation as a tourism booster**

The tourism industry is one of the best places for the application of machine translation. MT fulfills the communication requirement among people from diverse locations with different language and culture background. To make available the local information of any tourism place in multiple languages, MT system can help the tourists. There are countries like Macau, Palau in which the tourism revenue contributes to more than 60% of their gross domestic product (GDP). So to promote tourism in any country, it requires the efficient machine translation systems of tourism domain available at low cost [21].

1.2.2 Challenges in developing MTS

Processing of any natural language by the computer system has never been an easy task. Although in recent years the development of technology has increased the computational power of computer systems to a greater level, but still computers are not capable of handling natural languages as being handled by humans. The main reason for this is the presence of ambiguities in natural language. The following are the problems faced for developing MTS:

(i) **Ambiguity at Word level**

Lexical item is the word with its grammatical attributes and meaning. The ambiguity at word level occurs when one word has more than one equivalent lexical item. For example the word 'plane' may indicate a plane surface in the sentence

"They reached at plane area on Monday"

and indicates an airplane machine in the sentence

"The plane was flying over the hills".

(ii) **Ambiguity at syntax level**

It is an ambiguity in which the phrases are having more than one association among them in a sentence. For example in sentence "He saw a tiger on the hill with a telescope" , the phrase "with the telescope" is having multiple associations with the other phrases in the sentence as it can be associated with "a tiger", "the hill" or "saw".

(iii) **Ambiguity at semantic level**

Such ambiguity arises due the diverse contextual elucidation of one phrase in respect to other phrase.

For example in the sentences below, the word "kill" has different interpretations depending upon the context in which it is used:

(a) Ram killed Raavan.

(b) Ram killed the process with a system call.

In sentence (a) "kill" refers to the life of Raavan, a human and in (b) "kill" refers to the software program termination.

(iv) **Ambiguity at pronoun reference level**

One of the major problems in machine translation is the anaphora resolution or the pronoun reference resolution in the natural language sentences.

For example, in the sentences given below:

(a) The computer outputs the data, it is faster.

(b) The computer outputs the data; it is stored in Unicode.

In sentence (a) 'it' refers to 'the computer' whereas in (b) 'it' refers to 'data'.

Some solutions to this problem were provided by [22].

(v) **Free Word order Problem**

English language follows the word order Subject Verb Object (SVO), but languages like Sanskrit and Punjabi do not follow any particular word order. For example, all the sentences below are acceptable in Sanskrit, giving same meaning as output but have different word order Subject Object Verb (SOV), SVO etc.

रामः विद्यालयम् गच्छति

रामः गच्छति विद्यालयम्

विद्यालयम् रामः गच्छति

गच्छति रामः विद्यालयम्

The English equivalent of the above sentence will be Ram goes to school.

Such free word order makes more difficult to make word to word translation [23] for such languages.

(vi) **Ambiguity at pragmatic level**

Such ambiguities took place due to the pragmatic context of the sentence and this can only be resolved by taking into consideration the context of the user [23] . For example, in the sentence below:

Ravi hit the dog with a stick.

The ambiguity can be resolved only by taking into consideration the previous texts related to Ravi that whether Ravi has the stick and hit the dog in self-protection or the dog had a stick and hit Ravi. One solution for such ambiguity could be the world knowledge or the context knowledge or both [23].

1.3 Sanskrit Enconversion

It is the process of converting the Sanskrit text into an intermediary language representation i.e. UNL expressions which could be easily understood by computers and could be

used to generate the target language from that representation automatically. Here UNL expressions have been used as interlingua representation. Analysis of the Sanskrit language, Sanskrit-UNL dictionary and UNL expression generation rules are the basic requirements for developing Sanskrit Enconverter system.

1.4 Universal Networking Language (UNL)

UNL is the language of computers which enable them to process the information and knowledge of natural languages in a similar manner as human beings processes. With the help of UNL, the computers can communicate with each other with high speed and produces less amount of errors in processing any language text. UNL provides the linguistic infrastructure to process the natural language in computers and covers maximum components available in the natural language. The main objective of UNL program is to provide a platform over which people all around the world could communicate with each other in their native languages without any language barrier.

1.4.1 UNL versus other MT approaches

In 1996, the launch of a program as Universal Networking Language (UNL) in Tokyo Japan by the Institute of Advanced Studies (IAS) under United Nations University (UNU) was one of the big steps taken by researchers in the direction of MT development.
Several approaches have been used for developing MTS for different language pairs. Table 1.1 shows the comparison of MT approaches based on well defined parameters.

Table 1.1: Comparison of MT Approaches based on several criterion

MT Approach Criteria	Direct MT	Rule Based MT	Corpus Based MT	Neural MT
Morphological Analysis	Required	Required	Required	Done by Encoder
Syntactic and Semantic Analysis	Not Required	Required	Required	Encoder performs this task
Deep Linguistic Knowledge	Not Required	Required	Not Required	Training of Encoder and Decoder is required
Simple to Implement	Yes	No	Simple than RBMT	not simple,but less space is required than SMT
Cost	Less Costly	Costly in terms of Time	Costly in terms of Resources	costly in terms of computational power required(Needs GPU)
Fast Development	Yes	time Consuming	Faster Then RBMT	Once trained gives output in fractions of seconds
Efficiency	Better for simple and small translation	Most efficient	Better than DMT	better than SMT
Large Computation Required	No	No	yes	Yes
Word Level translation	Yes	Yes	Yes	No
Sentence Level Translation	No	No	Yes	End-to-End translation

The UNL system was based on Interlingua approach which is sub-part of RBMT approach. Enconverter and Deconverter are the basic tools required to develop MT in the UNL system for any natural language (NL). The former is used to convert the sentences from the NL to an Interlingua code known as UNL expressions and the later is used to do

the reverse i.e. from UNL expression to Natural Language. For n number of NLs, the UNL system requires development of only $2n$ components. The UNL interlingua is the world-wide recognized and standardized interlingua representation. UNL expressions have three basic building blocks as: Universal Word (UW) to represent the NL concepts, UNL relations to represent the role of each word in the NL sentence and UNL attributes to represent the semantic of the NL sentence. To resolve the word ambiguity in UNL there are 57 UNL relations which are continuously updated by the UNL foundation. The UNL system also uses the word dictionary known as UW dictionary.

Unlike other methods, UNL has several other applications than machine translation. UNL can be used to develop question answering systems, text summarizing, sentiment analysis, web browser development etc [24] . UNU in 2001 established an independent body to monitor the work of developing UNL program all over the world and named it as Universal Networking Digital Language (UNDL) Foundation.

1.4.2 UNL System

The UNL System comprises of three main components as: Language resources, Software to Process Language resources (LPS) and tools to develop new software for processing the resources or operating and maintenance of LPS. The language resources are further divided into Language Dependent (LD) and Language Independent (LI) parts. The LD part consists of dictionaries and analysis/generation rules which are stored on language servers (LS). Various LS communicate with each other using internet or can be accessed via internet from anywhere in the world. The LI part consists of UNL Knowledge Base (UNLKB) and UW Dictionaries which are maintained and developed by supporting tools locally or through internet. The UNLKB describes the UNL concepts and their relations. The software processing tool like UNL verifier which verifies the UNL expressions generated by supporting tools through internet and it is also connected to various LS with internet.The language server (LS) consists of two main components : Enconverter and Deconverter. EnConverter is the mechanism

which converts the natural text into UNL expressions using dictionaries and the translation rules. Deconverter converts the UNL expressions into Natural language text. IAS of United Nations University has developed software for enconversion and deconversion as EnCo and DeCo wihich are available online with dictionary, conversion rules and co-occurrence dictionaries for specific languages. Supporting tools act as UNL editor and generates the UNL expressions in connection with Language servers (LS) via internet. Figure 1.2 shows the architecture of UNL system.

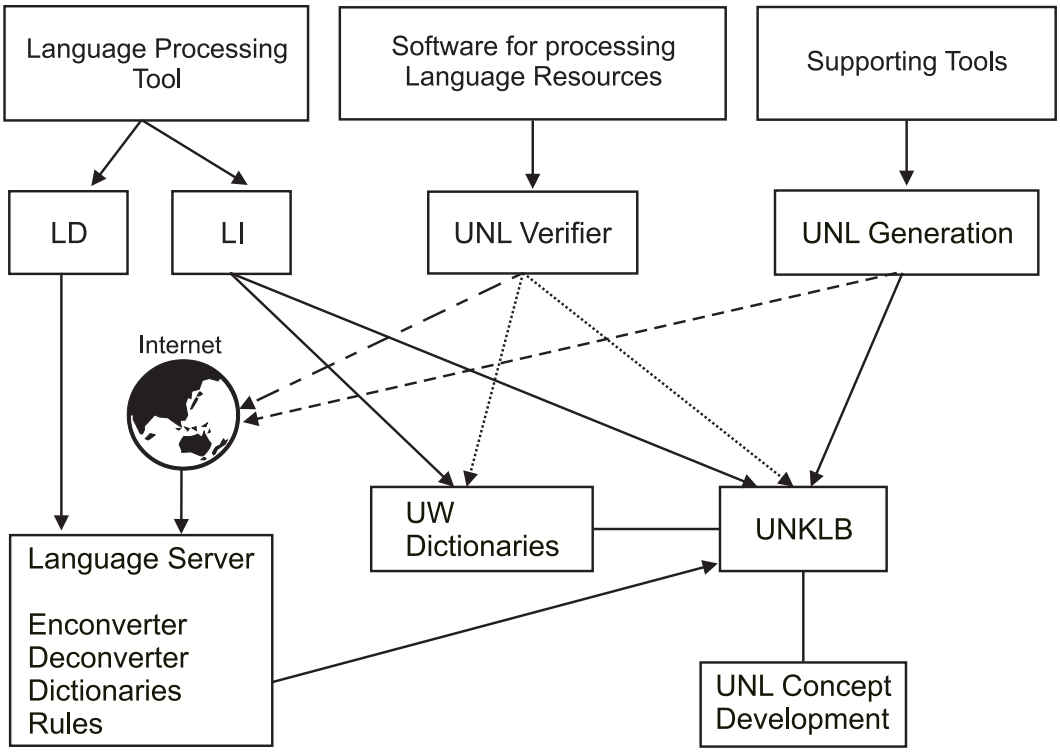

Figure 1.2: Architecture of UNL System

Software for processing the language resources like UNL Verifier which verifies the UNL expressions generated by UNL generator/editor tool with the help of UNLKB, UW dictionaries and language servers [25].

Representation of Information in UNL

UNL represents the information in the form of a semantic network which is a labeled directed graph. The nodes of graph represents concepts known as UW [26] and edges represents UNL relations are used to find role of each word in the sentence. The third component is UNL

attribute are used to express author's subjective meaning of the sentence [27].

(a) **Universal Word (UW) First Component of UNL**

UW is the basic backbone of UNL based translation system. UW signifies the concepts in natural language. It is made up of string of characters followed by constraints list [28] as shown below:

```
<UW>::=<headword>[<constraint list>]
```
(1.1)

where <headword> is an English word which denotes the concept in English language equivalent to source language.

There are four types of UWs [27] as given below:

(i) Basic UWs are the words without constraint list.

For example: go, take, house etc.

(ii) Restricted UWs are the UWs with constraint list.

For example: the word '`state`' without constraint list may represent several concepts in English like:

$$state(icl > Country)\ represents\ the\ country.$$

$$state(icl > region)\ represents\ a\ region.$$

$$state(icl > government)\ represents\ a\ government.$$

$$state(icl > abstractthing)\ represents\ the\ thread\ state\ or\ a\ process\ state.$$

Without putting the constraints in brackets the '`state`' word would be ambiguous in nature. The constraint list helps in removing the ambiguity of the word and restricts it to a particular concept.

(iii) Extra UWs are the UWs which do not have their equivalent word label in English language. For example

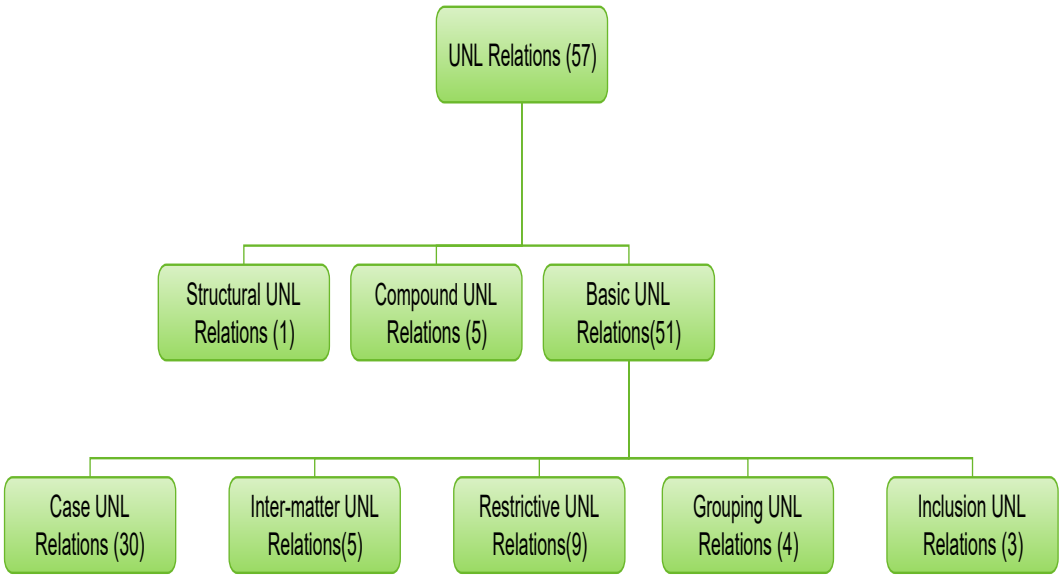

Figure 1.3: Hierarchical classification of UNL Relations

`Kalpa(icl>duration)`

(iv) Temporary UWs are the words which do not have any definition and are used directly in the UNL document to represent any number, e-mail id or URL.

(b) **UNL Relations**

The next major component in UNL is UNL Relation. UNL expressions are formed with the help of UNL binary relations and UWs as:

$$\texttt{rel(UW1, UW2)} \tag{1.2}$$

where `rel` is the UNL binary relation and `UW1`, `UW2` are the Universal Words which are having '`rel`' relation among them.

Figure 1.3 shows the classification of UNL relations. UNL have 56 relations for semantic information extraction and one for the structure identification relation. The basic UNL relations have further been divided into five parts. Table 1.2 shows 30 case identification UNL relations, Table 1.3 shows five inter-matter, Table 1.4 presents nine restrictive, Table 1.5 presents four grouping relations and three inclusion relations are shown in Table 1.6. There are five compound UNL relations and one structural

relation as shown in Table 1.7.These relations are used to cover all the aspects of natural language and identifies the thematic role of words in a sentence [29].

Table 1.2: Case Identification UW

Sr. No.	Relation	Basic Component	Description	Example
1	agt	Agent	Designate a thing which initiate an action	Ravi breaks the rule agt(break, Ravi) Rama killed Ravana agt(kill, Rama)
2	aoj	Thing with attribute	Specifies a thing which has an attribute or in a state	John is an teacher aoj(teacher, John) He knows about this incident aoj(know, He) She has a computer aoj(has, She)
3	bas	Basis	Specifies a thing as basis of comparison	Ram is more intelligent than Shyam man(intelligent, more) bas(more, Shyam) She will buy the same bicycle as you bas(same, you)
4	ben	Beneficiary	Specifies a victim or beneficiary of an event/ action/ state	Daksh gives a flower to Sunita ben(give, Sunita) The soldier dies for the nation ben(die, nation)
5	cag	Co-Agent	Defines a thing which is out of focus and done in parallel due to an implicit start of event	Rahul walks with Hitesh agt(walk, Rahul) cag(walk, Hitesh) Garima lives with her uncle agt(live, Garima) cag(live, uncle)
6	cao	Co-Thing	Defines an out of focus thing that is in parallel state	The tiger was white with black spots cao(white, spot) mod(spot, black)
7	cob	Affected Co-thing	Defines a directly affected thing by an implicit action done in parallel or an occurrence of a parallel state	Harish died with Hitesh cob(die, Hitesh) obj(die, Harish)

Table 1.2 Continued: Case Identification UW

8	dur	Duration	Defines an interval of time during which a state or an event occurs	Smith works ten hours a day agt(work, smith) dur(work, hour) qua(hour, 10)
9	exp	Experiencer	Defines an experiencer which is indirectly involved in a state or an action	She had her laptop stolen exp(steal, she)
10	gol	Goal/ Final state	Defines the final state of a thing or object finally associated with the object of an event	The signal light changes from green to red gol(change, red) most of the black money is deposited into Swiss bank account gol(deposit, account)
11	ins	Instrument	Defines the instrument to perform an event	he writes on black board with a chalk ins(write, chalk) he cut the cake with a knife ins(cut, knife)
12	man	Manner	Defines the manner in which an event is carried out or state characteristics	most beautiful man(beautiful, most(icl>how) he often visit Delhi man(visit(agt>thing, obj> thing), often)
13	mat	Material	Defines the material with which the thing is made	mango juice mat (juice (icl>liquid), mango(iof>fruit)) golden ring mat(ring (icl>obj), golden(icl>metal))
14	met	Method/ Means	Defines a method to carry out an event	he solves the problem with algebra met(solve(icl> resolve (agt>thing, obj>thing)), algebra(icl>method)) he separates the edges by cutting met(separate,cut)
15	obj	Object	Defines the focused object of a state/ event/ art	He goes to school

Table 1.2 Continued: Case Identification UW

				obj(go(icl>thing), school(icl>place))
				She have a pen drive
				obj(have(aoj>thing, obj>thing), pendrive(icl>thing))
16	opl	Affected place	Defines a focused place affected by event	The cake was cut in the middle
				opl(cut (agt>thing, obj>thing, opl>thing), middle(icl>place))
				she, pats her daughter on the arm
				opl(pat(icl> touch(agt> thing, obj>thing, opl>thing)), arm(pof>trunk))
17	plc	Place	Defines the place of occurrence of an event or existence of a thing or state truth	She cooks in kitchen
				plc(cook(agt>thing), kitchen(pof>building))
				It's hot here
				plc(hot(icl> temperature), here(icl>place))
18	plf	Initial place	Defines the place of beginning of an event or state becomes true	He came from London
				plf(come (icl>thing), London(icl>city))
19	plt	Final place	Defines the place of end of an event or place where state becomes false	My friend went to London
				plt(go(icl>do), London(icl>city))
20	ptn	Partner	Defines the crucial out of focused initiator of an action	Seena competes with Shan for job
				ptn(compete (agt>thing, ptn>thing), Shan(iof>person))
				Ravi celebrates his birthday with poor people
				ptn(celebrate (icl>thing), poor people(iof>person))
				She collaborates with Ram
				ptn(collaborate (agt>thing>, ptn>person), Ram(iof>person))
21	qua	Quantity	Defines the quantity of a thing or quantity of work done	He studies nine hours a day for IAS examination
				qua(hour(icl>period),9)

Table 1.2 Continued: Case Identification UW

				per(hour(icl>period), day(icl>period))
				Two students
				qua(student(iof>person),2)
22	rol	Role	Defines a role of a thing	Ravi works as General Manager in the company
				rol(GeneralManager (icl>position), Ravi(iof>person))
23	scn	Scene/ Field	Defines the scene of occurrence of an event or existence of a thing	He won the first prize in a fair competition
				scn(win(icl>thing), competition(icl>event))
				...to come on a radio program...
				scn(come(gol>thing, obj>thing), program(icl>plan))
24	soj	Thing in a state	Defines a state of a thing	how someone can survive in this flood
				soj(how, someone)
				at last they left something in good condition
				soj(condition (icl>state), something)
				cancer patient in danger of dying
				soj(danger(icl>possibility), patient(iof>person))
25	src	Source/ Initial State	Defines the source of an object or thing associated initially with the object of an event	As soon as signal changes from red to green the traffic speed increases
				src(change(obj>thing), green(icl>color))
				She quickly removed her hands from the gas flame
				src(remove(agt>thing, obj>thing), gasflame(icl>thing))
26	sta	State	Defines the Object state during its operation	dinner eaten cold

Table 1.2 Continued: Case Identification UW

				sta(eat(agt>thing, obj>dinner), cold(aoj>dinner))
27	tim	Time	Defines the event occurrence time or time when a state is true	She was forced to leave on Monday
				tim(leave(agt>thing), obj>place), Monday(icl>day))
				she starts to do her home work at 9:00 o'clock
				tim(do(agt>thing, obj>thing), o'clock(icl>time))
28	tmf	Initial Time	Defines the starting time of an event or when a state is true	Workers in company work from Monday to Saturday
				tmf(work(agt>thing), Monday(icl>day))
				Her nature has changes since she came here
				tmf(change(obj> thing), come(icl>thing))
29	tmt	Final Time	Defines the ending time of an event or time when a state changes to false	Workers in company work from Monday to Saturday
				tmt(work(agt>thing), Saturday(icl>day))
				She completes her work on Monday
				tmt(complete(agt>thing, obj>thing),Monday(icl>day))
30	via	Intermediate Place/ State	Defines the transitional state/ place of an event	he reached Delhi via Surat
				via(reach(agt>thing, obj>thing), Surat(icl>city))

Table 1.3: Intermatter UNL Relations

Sr. No.	Relation	Basic Component	Description	Example
1	con	Condition	Defines a non-focused state/ event that conditions a focused state or event	If the light is red, then you cannot go
				aoj:01(red,light)

Table 1.3 Continued: Intermatter UNL Relations

				con(go, :01)
				agt(go, you)
2	coo	Co-occurrence	Defines a parallel state or event for a focused state or event	Baby was crying while running
				coo(cry, run)
				The air was cold as well as slow
				coo(cold, slow)
3	pur	Purpose	Defines the objective of the agent of an event or existence of a thing	he went to university for research
				pur(go(icl>do), research(icl>do))
				He works for money
				pur(work(icl>do), money)
4	rsn	Reason	Defines the reason behind the occurrence of an event or state	He did not play due to health issues
				rsn(play(icl>act), health issue(icl>thing))
				Jaipur city is known for its beauty
				rsn(known(aoj>thing), beauty(icl>abstract thing))
5	seq	Sequence	Defines a prior event or state of focused event or state	Think before you speak
				seq(speak(icl>do), think(icl>do))
				Signal was red and then green.
				seq(green(icl>color), red(icl>color))

Table 1.4: Restricted UNL Relations

Sr. No.	**Relation**	**Basic Component**	**Description**	**Example**
1	cnt	Content	Defines the content of concept	SANSUNL, Sanskrit Enconverter will succeed
				cnt(SANSUNL, Sanskrit Enconverter)
2	deg	Degree / Level	Defines the level or the degree of a thing	level of quality
				deg(quality, level)
				a degree of slope

Table 1.4 Continued: Restricted UNL Relations

				deg(slope, degree)
3	mod	modification	Defines a thing which restricts a focused thing	She will tell the whole story
				mod(story, whole)
				David is more intelligent than Shyira
				mod(intelligent, more)
4	mof	member of	Defines the membership of the concept with the collective concept	employee of Infosys
				mof(employee(icl>person), Infosys(icl>organization)
				member of parliament
				mof(member, parliament)
5	ori	Original	Defines the originality of a thing	You should take back-up copies of your project
				ori(copy(icl>thing), project(icl>thing))
6	nam	Name	Defines the name of an event	his father ”Ravi Sharma”
				nam(father (icl>relative), Ravi)
7	per	Proportion/ Rate / Distribution	Defines a unit or basis of distribution/ rate / proportion	He studies nine hours a day for IAS examination
				per(hour(icl>period), day(icl>period))
				qua(hour(icl>period), 9)
				He visits Delhi twice a week
				per(time(icl>frequency), week(icl>period))
8	pof	part of	Defines a concept the focused event is part of	EnCo is part of Language server
				pof(Enco(icl>thing), Language server (icl>thing))
				car’s engine
				pof(engine(icl>machine), car(icl>vehicle))
9	ran	Range	Defines a range of a thing	university campus has become place of activities
				ran(activity(icl>action), place(icl>place))

Table 1.5: Grouping UNL Relations

Sr. No.	Relation	Basic Compo-nent	Description	Example
				Sita and Gita
1	and	Conjunct	Designates a partner having conjunctive relation to	

Table 1.5 Continued: Grouping UNL Relations

				and(Gitaperson), Sita(iof>person))
				Rama and Krishana
				and(Krishana(iof>person), Rama(iof>person))
				eating and talking
				and(talk(icl>do), eat(icl>do))
2	a/o	And/ Or	Specifies the partner to have and/or relation to	singing and or dancing
				a/o(dance(icl>act), singing))
3	int	Intersection	Defines the common properties/ instances with the partner concept	intersection of national highway and state highway
				int(national highway (icl>abstract thing), state highway (icl>abstract thing))
4	or	Disjunction	Defines disjunctive relations between two UWs	the cat is black or brown
				or(brown(icl>color), black(icl>color))
				who is going to Delhi, Ravi or Ram
				or(Ramperson), Ravi(iof>person))

Table 1.6: Inclusion Relation

Sr. No.	**Relation**	**Basic Component**	**Description**	**Example**
1	equ	Equivalence	Defines an equivalent concept	An Enconverter (language encoder)
				equ(Enconverter, language encoder)
2	icl	Included in/ a kind of	Defines the generalized concept	a human is a kind of animal
				icl(human, animal)
				a gameof football
				icl(football, game)
3	iof	an instance of	Defines the class concept to which the instance belongs to	Delhi is a city in India
				iof(Delhi(icl>city, city in India))

Table 1.7: Compound Relation

Sr. No.	Relation	Basic Component	Description	Example
1	fmt	range/ From-to	Defines a range between two things	She can learn the alphabets from a to z
				fmt(z, a)
				The distance from Delhi to Udaipur if more than 600 km
				fmt(Udaipur(icl>city), Delhi(icl>city))
2	frm	Origin	Defines the initial state or origin of the focused event	a scientist from ISRO
				frm(Scientist(icl>person, ISROgroup))
3	aut	Author	specifies the producer/ author/ creator of thing	a statement by the leader
				aut(statement(icl>thing), Leader(icl>person))
				Sita's writings
				aut(writing(icl>thing), Sita(iof>person))
4	pos	Possessor	Defines the owner of a thing	Ravi's bike
				pos(bike(icl>thing), Ravi(iof>person))
				My laptop
				pos(laptop(icl>computer),I)
5	to	Destination	Defines the association of final state of a thing with focused thing	The players packed their baggage for Mumbai
				to(baggage(icl>luggage), Mumbai(icl>city))
				She wrote a letter to administration
				to(letter(icl>document), administration(icl>person))
Structural Relation	shd	Sentence head	Defines a mark to show the location of word/ sentence/ paragraph in a book/ document	Chapter 1 Introduction
				shd(Introduction(icl> state), Chapter(pof>book)
				mod(chapter(pof> book),1)

(c) **UNL Attributes**

The third component of UNL system is UNL attribute. The subjective knowledge of the sentence is described by UNL attributes. The attributes expresses the following:

- The view point of the speaker
- The concept range as generic or specific
- To express the logical expressions

There are eight categories of UNL Attributes as shown in Table 1.8.

Table 1.8: UNL Attributes

Describing UW's Logicality	
@transitive	to show transitivity property among UW and attached to the ancestor UW
	(If A precedes B and B precedes C then, A precedes C is true)
@symmetric	part of the UW having symmetricity
	If A is brother of B, then B is also brother of A. So @symmetric is attached to brother
@identifiable	Associated with the UW which identify the subject
@disjointed	associated with the UW which shows the disjoint property
	this can be associated with single UW of the scope to show that members are not allowed to share common concepts
Attributes telling times with respect to Speaker	
@past	happened in past
	It was raining yesterday
@present	happening at present
	It is raining hard
@future	will happen in future
	The schools will open on Monday
Attributes focusing on aspects of event from speaker's view	
@begin	indicates the beginning of an event
@complete	indicates the completion of an event
@continue	indicates the continuation of an event
@custom	indicates the customary action
@end	indicates the termination of an event
@experience	indicates the experience
@progress	indicates the event is in progress
@repeat	indicates the repetition of an event
@state	indicates the final state
@just	indicates an event has just begun or ended
	He has just come. come .@complete .@just

Table 1.8 Continued: UNL Attributes

@soon	indicates an event is about to begin/end
	The plane is about to take-off
@yet	indicates the event has not yet started or ended
Attributes telling of Speaker's view of reference to concept	
@generic	Indicates the generic concept
@def	indicates the already referred concept
@indef	indicates the non-referred concept
@not	indicates the negation
@ordinal	indicates the ordinal numbers
Attributes showing speaker's emphasis, focus and topic	
@contrast	represents contrasted UW
@emphasis	represents emphasized UW
@entry	represents the main UW of scope or the sentence
@qfocus	represents the focused UW in interrogative sentences
@theme	represents the theme of the sentence
@title	represents the title
@topic	represents the topic of discussion
Attributes showing speaker's attitude	
@affirmative, @confirmation, @humility, @exclamation @imperative, @interrogation, @invitation, @request, @respect, @vocative	
Attributes showing speaker's feeling, judgement and viewpoint	
@ability, @admire, @conclusion, @consequence, @blame @dissent, @grant, @grant-not, @although, @discontented @expectation, @wish, @insistence, @intention, @will @need, @obligation, @obligation-not @should, @certain, @may, @inevitable, @possible @probable, @may, @rare, @unreal, @regret, @surprised	
Convention	
@pl, @$angle_bracket$, @brace, @$double_parenthesis$, @$double_quote$ @parenthesis, @$single_quote$, @$single_bracket$	

(d) **UNL Expression**

To represent the information in a natural language sentence, UNL expression uses directed semantic graphs where nodes of the graph represents the concepts and arcs represents the relation among the concepts. The nodes of the graphs are annotated with UNL attributes to remove ambiguity in the sentence. A scope in UNL expression represents a hyper–node which in turn represents another semantic graph of a complex natural sentence and is denoted by scope-id in the UNL expression and ranges from

00 to 99. UNL expressions are unambiguous in nature. UNL expression may be represented either in tabular form or in list form as follows [29] (http://www.undl.org/unlsys/unl/unl2010/):

(a) **Table Form representation of UNL expression:**

{unl}

<UNL Relation>

.....

.....

{/unl}

<UNL Relation>::=<relation>[“:”<Scope-ID>]

“(“{{<Source UW>[<attribute list>]“,”|

{{<Target UW>[<attribute list>]")"

<attribute list>::={"."<UNL attribute>}.....

To disambiguate UWs further unique UW-id’s are used which ranges from ‘0-9’ and ‘A-Z’.

(b) **List form of UNL Expression:**

{unl} [W] <{<UW> | {":"<Scope-ID>}}[<attribute list>]":"<UW-ID>

.....

[/W]

[R]

<UNL relation hold by UW-ID>

.....

[/R]

{/unl}

where W represent the UW list and R represents the UNL relation and acts as HTML tags i.e. open and closed tags.

<UNL relation hold by UW-IDs>::

=<UW-ID1> <UNL relation[":"<Scope-ID>]<UW-ID2>

1.4.3 EnConverter

It is a tool which performs the task of analyzing natural language text at morphological, syntactic and semantic level. It converts the natural language text into UNL expression. Input to this tool is a natural language sentence in the form of strings. EnConverter tool performs a language independent parsing of input sentence with the help of word dictionary (L-UW) and set of Enconversion rules for a specific language. By changing the word dictionary and Enconversion rule, same tool can also be used to do analysis of a different language. In Enconversion, first step is tokenization of input string with the help of the word dictionary to get set of morphemes. Then using Enconversion rules syntactic and semantic graphs are generated of the morphemes and then the UNL expression is generated from the semantic graphs. The Enconverter tool consists of two windows: Analysis Window (A) and Condition Window(C). The Analysis Window takes input from the node list (morphemes generated during the parsing phase) one by one and the Condition Window checks the applicability of the UNL Enconversion Rule. Each node from the node list is processed by the 'A' window and 'C' Window and finally the syntactic and semantic graphs are generated by the Enconverter tool for corresponding input sentence. Enconverter can refer to UNL Ontology to remove ambiguity among words (http://www.undl.org/unlsys/unl/unl2010/).

1.5 Thesis Organization

The work done in this thesis has been organized in six chapters. An overview of these chapters is represented as follows:

Chapter 1 presents the introduction, need and challenges faced in machine translation. Features of Sanskrit language have also been discussed in this chapter. UNL structure with different components have been elaborated. Further, the Sanskrit Encoverter overview has

also been discussed. Finally thesis organization and contribution has been highlighted.

Chapter 2 performs a systematic literature review of various machine translation systems, linguistic tools, data repositories and MT platforms. The historical evolution of machine translation along with comparison of MT development approaches has also been included in this chapter. Based on the outcome of this survey, the research gaps have been identified and then the problem statement has been formulated followed by the research objectives.

Chapter 3 shows the proposed system named "SANSUNL". The SANSUNL system consists of seven layers such as pre-processing, POS tagging, parsing, node list creation, case marker identification, unknown token handling and UNL generation. All these layers have been explained in detail in this chapter.

Chapter 4 presents the implementation of the proposed system. The structure of Sanskrit-UW dictionary and format of enconversion rule-base along with all data-sets used for implementation has been highlighted in this chapter. The implementation of the proposed system is demonstrated using two examples. At the end of the chapter, Sanskrit to English MT system has also been added to show the application of proposed system.

Chapter 5 shows the testing and evaluation of the proposed SANSUNL system. For testing of the proposed system five data-sets have been used. Fluency score, adequacy score and BLEU score methods have been used to evaluate the proposed system.

Chapter 6 presents conclusion of thesis along with future scope of research.

1.6 Thesis Contributions

The contribution of this thesis work is of many folds, some of the important points are listed below:

(i) This thesis presents a description of Sanskrit language in a systematic way and its importance.

(ii) This work presents the UNL system, its components and comparison with other approaches.

(iii) The historical development of MTS and the MT approach based literature review of the existing MTS is presented in this thesis.

(iv) The language divergence among Sanskrit and English is discussed and possible solution are presented.

(v) The problem of Sanskrit to UNL Enconversion is addressed well and a novel solution is presented.

(vi) Machine learning based Sanskrit POS tagger is designed, implemented and tested.

(vii) This work presents a novel algorithm to generate parse tree from CYK parsing table.

(viii) This work presents a hybrid approach for Sanskrit to English translation.

Chapter 2

Literature Review

The desire of making automatic and fast translation using computer machines has been the biggest fantasy of human beings. Researchers have worked all around the world to fulfill this desire of developing automatic MT systems. The aim of this chapter is to present research work associated with historical evolution of MTS, discussion and comparison of various MT approaches, development approach based MTS classification and available MT platforms and tools. In order to understand the concept of MT systems, extensive literature has been collected from various resources that includes major books, review papers, research papers, magazines, technical reports, symposiums etc. An attempt has been made to include all relevant research papers to perform comprehensive survey. This chapter also describes the research gaps generated from comprehensive analysis of existing research work and also includes problem formation. At last, the research objectives are listed.

2.1 Historical evolution of MTS

The task of translation started even before computer machines came into existence. The idea of machine translation came into picture in 17th century when Discartes and Leibniz proposed the concept of mechanical dictionaries based on the method of universal numerical codes. Cave Beck, Athanasius Kircher and Johann Becher in middle of the 17th century has presented actual implementation of mechanical dictionaries proposed by Discartes and Leibniz.

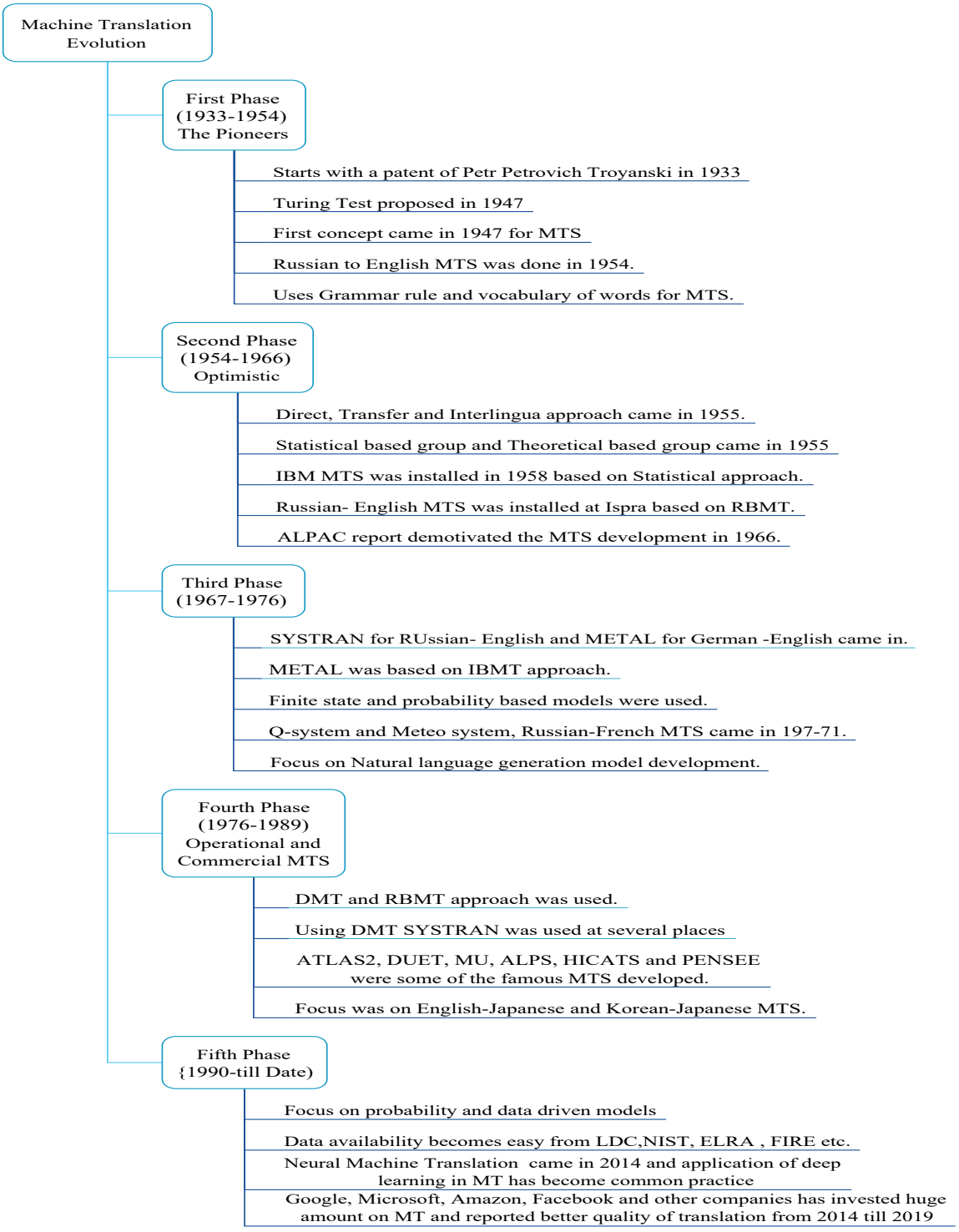

Figure 2.1: MT Evolution in General [1, 2, 3, 4, 5]

Development of an unambiguous language for these dictionaries on basis of universal language movement, logical principles and iconic symbols has been the motivation behind

this work.

The actual machine translation took place in 20th century. The history of Machine translation could be divided into five phases as shown in Figure 2.1. Researchers all around the world proposed different machine translation systems based on various approaches.

Based on the history, machine translation has been divided into four categories [2, 30] as discussed below:

- Machine Translation for Watcher

  As the name indicates such type of machine translation is used by persons who want to access the information encoded in foreign languages and are ready to accept even the rough output of the translation system. Such systems are imagined by the Pioneers and are based on bilingual dictionary matching. SYSTRAN is an example of such machine translation systems used to translate military technical texts.

- Machine Translation for Revisers

  These systems are automatic machine translation systems with the better quality which can be compared with drafts produced by human. LOGOS is an example of such systems. TAUM-METRO is the first MTS used to translate the Canadian weather reports into French.

- Machine Translation for Translators

  These types of systems are used by the human translators to make their translation job easy with the help of online databases, lexicons and the translation memories. SISKEP, an English-Malay MTS is an example of such machine translation systems. IBM's translation Manager; Trados's TWB etc. are some of the online available tools for the professional translation.

- Machine Translation for Authors

  These are the systems used by the authors to translate their texts into other languages

and also to remove utterance ambiguity to get the better translation without the need of any further revisions. Such types of systems are interactive during analysis and transfer phase of the translation. The authors are not supposed to access the system, the specific experts performs the translation tasks and give the output to the authors. TITUS is an example of such systems which is used by the Textile Institute of France and N-Trans is another such system used by Japanese.

2.2 MT Approaches

Machine translation can be done by using various approaches. Vauquois Triangle shows various machine translation approaches in the Figure 2.2.

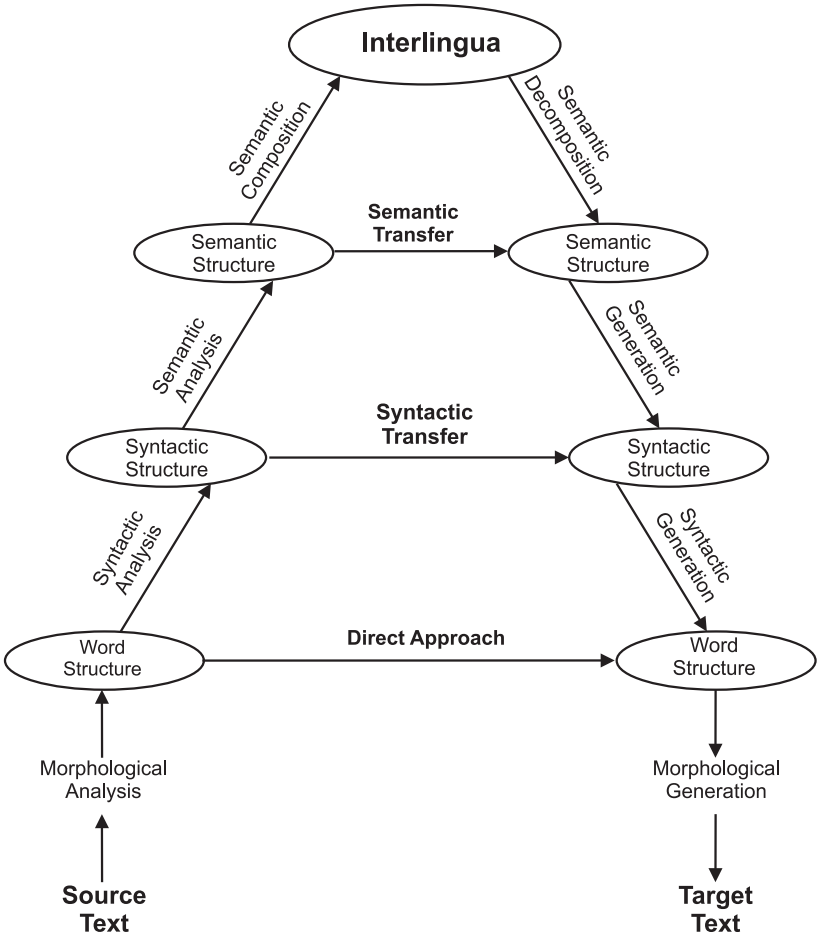

Figure 2.2: Vauquois Triangle

In Figure 2.2, if we go from bottom to top, the analysis complexity level is increasing.

Width of (triangle) i.e. the horizontal arrows are showing level of effort required for translation. At the bottom of triangle, word by word translation of text is performed and at the top, syntactic as well as semantic level analysis needs to be done for translation.

Further the classification of various MT approaches is shown using Figure 2.3. The detailed description of these approaches is given below:

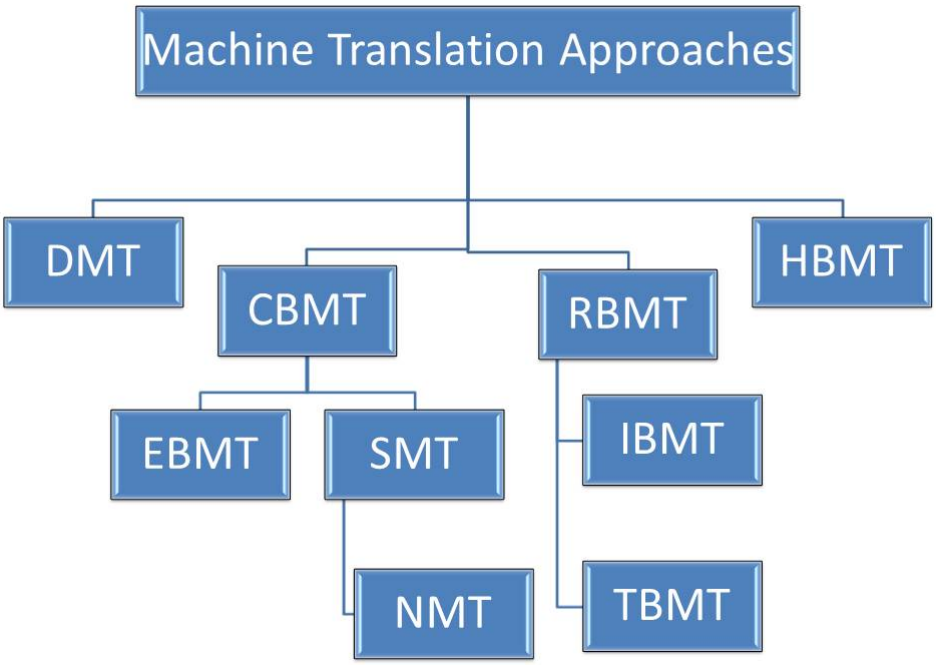

Figure 2.3: Machine Translation Approaches[6, 7]

(i) Direct Machine Translation (DMT)

This approach was established in 1950's for developing the machine translation systems using newly developed computer systems. For direct approach there is no intermediary representation of source and target language, only word to word matching is performed for translation.The system may have pre-processing for the input sentence morphological analysis and post-processing phases for target sentence reordering. The system uses a bilingual dictionary for matching the source language words with the target language words [6]. The process of direct translation is shown in Figure 2.4. This approach is used where there is little knowledge about the language analysis

syntactically as well as semantically for the source and target language [31].

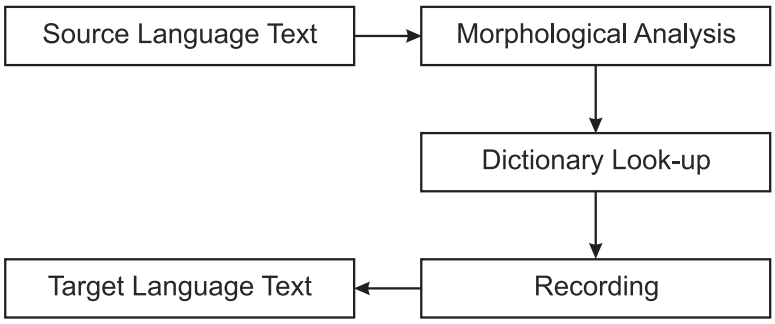

Figure 2.4: Direct Machine Translation Approach

Example 1. Ram goes to school रामः गच्छति विद्यालयम्

To translate sentence given in Example 1 using direct approach requires finding Sanskrit equivalent word for each English word using English- Sanskrit bilingual dictionary. The post editing is performed to arrange the word order by the target language. The word to word translation is shown above and as Sanskrit language is word order free language so there is no requirement of re-ordering of the word.

(ii) Rule Based Machine Translation (RBMT)

This approach is an enhancement to the direct machine translation approach. In this, source language text is analyzed first and then an intermediate code representation is generated which could be a language independent or dependent code from which the target language text is generated. As the name indicates, in this approach set of rule base are used in analyzing the source language text, intermediate code generation and the final target language text generation. Based on the intermediate code, this approach is further divide into two categories:

(a) TBMT

This approach has three components for the translation as analysis, transfer and synthesis. The Analysis component analyzes the source language text to get the syntactic information as well as the semantic information about the words

in the sentence. Based on the information retrieved in the previous phase, the intermediate code i.e. parse tree is generated. Synthesis component generates the target language sentence using the rule base from this intermediate code. Figure 2.5 shows TBMT approach [32].

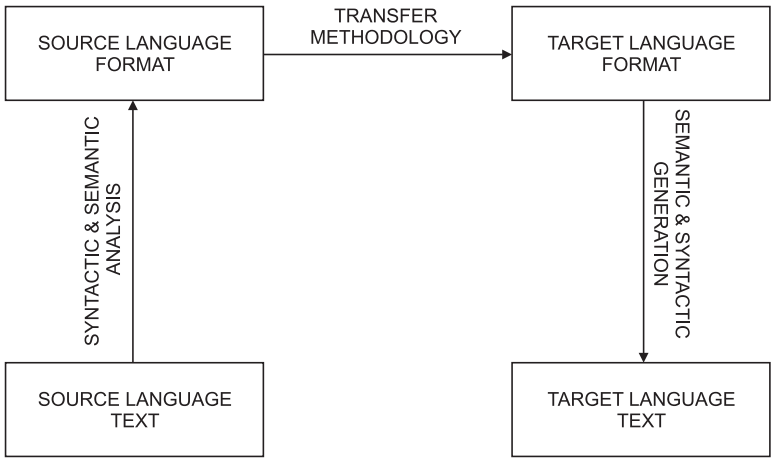

Figure 2.5: Transfer Based Machine Translation (TBMT)

The modularity of this approach makes it better than the DMT. The complexity in designing of rule base makes this approach difficult in a multilingual environment.

(b) Interlingua Based Machine Translation (IBMT)

This approach comes at the top of the Vauquois Triangle. In this approach, text from source language is analyzed and a language independent intermediate code is generated. Using this intermediate code, any target language could be generated. Due to the language independent intermediate code (Interlingua code) for any language, this approach requires only two components: one converting from source text to intermediate text and second for generating target text from the intermediate text. This approach best suits in multilingual environment. Figure 2.6 shows the IBMT approach [31].

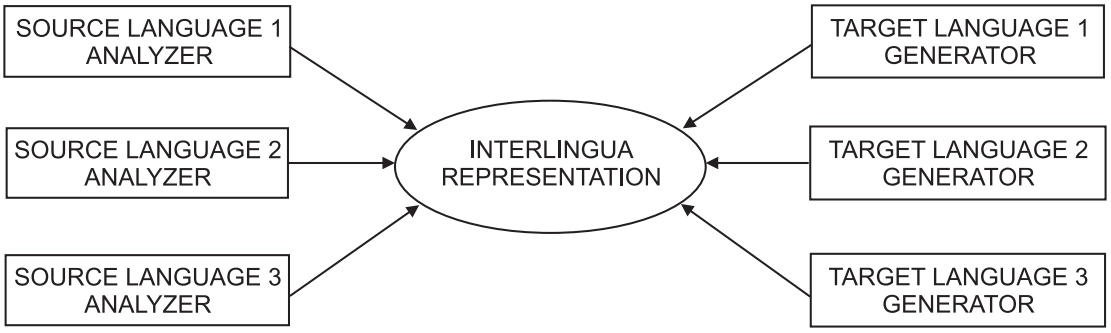

Figure 2.6: Interlingua Based Machine Translation Approach

(iii) Corpus Based Machine Translation (CBMT)

Due to large availability of the text in digital format, this approach has become more prevalent than other approaches. As the name indicates, this approach requires a corpus of parallel aligned source and target language sentences. If the corpora exist for the particular language pair then the efficiency and the speed of translation using this approach is better than other approaches. But this advantage is also a big disadvantage as this approach does not work for language pairs for which no parallel corpora exists. This approach further has two sub-divisions as follows:

(a) Statistical Machine Translation (SMT)

In this approach, statistical or probabilistic techniques has been applied in the machine translation system development. There are two major components of this approach : language model and translation model. The language model produces probability of occurrence for the strings of words in source as well as target language and also the conditional probabilities of occurrence of a word in target language which translates a word in the source language. The multiplication of probability of occurrence of a word in source language with conditional probability of occurrence of a word corresponding to this word in target language provides occurrence of source and destination pairs of words in corpus available for translation. This method requires large amount of database and very complex

statistical techniques to do the translation. The efficiency of the system increases with more training data sets and parallel corpora availability for the language pair. Machine translation can be done as word based, phrase based, sentence based or hierarchical phrase based. The translation model generally uses the n-gram model. N-gram model predicts the occurrence of the next word of the text given the previous words. Figure 2.7 shows the architecture of SBMT approach.

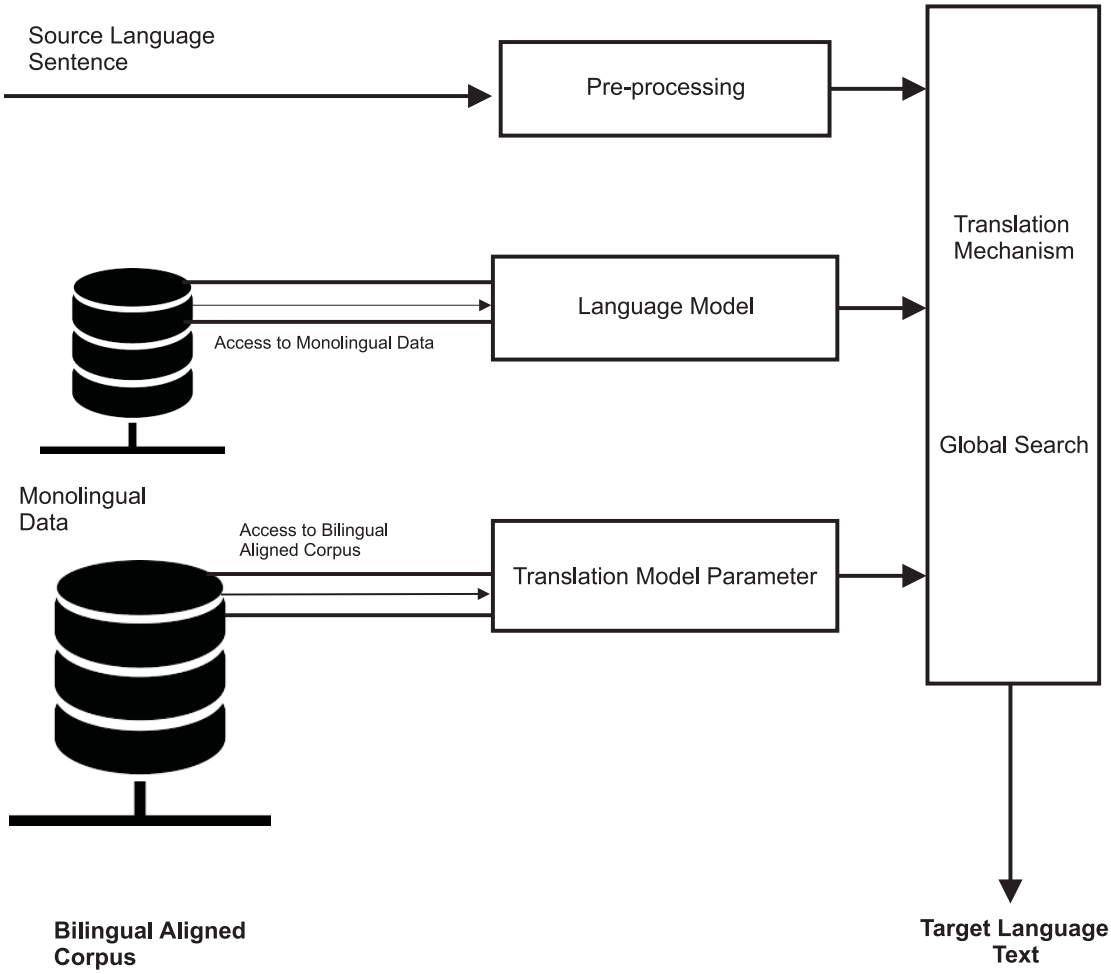

Figure 2.7: Statistical Based Machine Translation Approach

(b) Example Based Machine Translation (EBMT)

This approach was first introduced in 1984 by M. Nagao. The basic translation principle for translation used by this approach was analogy. This approach does

not require huge amount of corpora, it needs a bilingual corpus of stored examples and uses one of the matching algorithm to find the translation which best matches with the source language sentence at present [33]. Generally EBMT does not require any grammar rule base in detail; it uses only the stored examples and the matching algorithm to find the closest match corresponding to the given input sentence. Figure 2.8 shows the architecture of EBMT approach [6, 33].

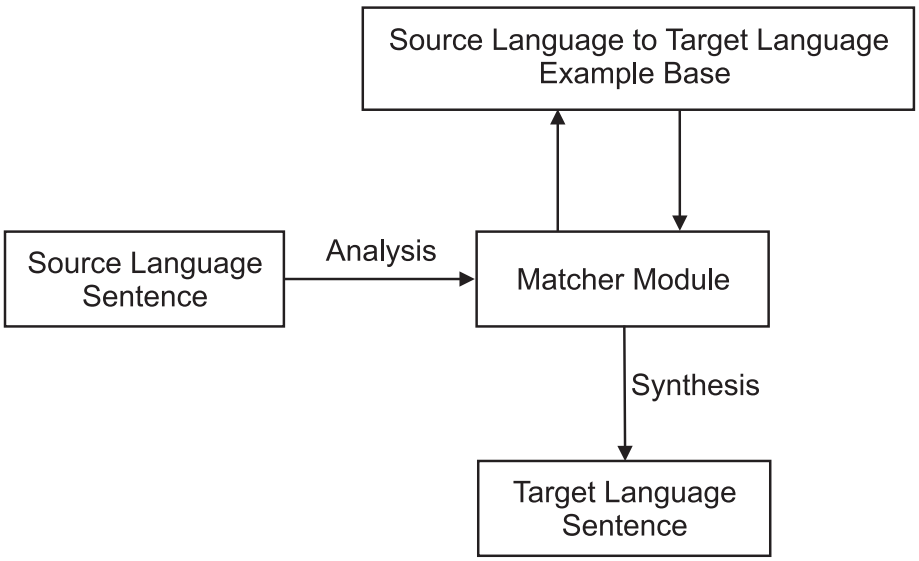

Figure 2.8: Example Based Machine Translation (EBMT) Approach

(iv) Neural Machine Translation (NMT)[3, 4, 5]

With the explosive growth of the internet and easy access to high computing power systems, NMT has emerged as a fast-growing approach for developing new MTS. Basic components of NMT system are encoder and decoder. It uses single neural network architecture to generate target sentence for the input sentence as shown in Figure 2.9. Initially, the problem with NMT systems was the fixed size vector space generated by the encoder for input sentence which was resolved by [34].

Different types of neural network architectures have been used for developing new MTS. Recurrent Neural Network (RNN) are used mostly for MTS development due to its feature of preservation with the processing of input data memorization. Long Short Term Memory (LSTM) is a type of RNN with two or more than two hidden layers

and is used for extracting features from input text and also increases the efficiency of translation [35].

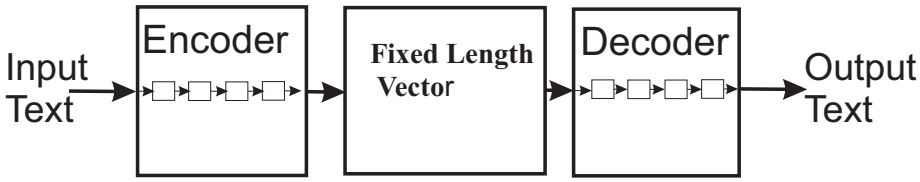

Figure 2.9: NMT System Architecture

2.3 Development approach based MTS classification

This section provides literature review of various MT systems on the basis of approach used for development of that system.

2.3.1 DMT approach based systems

(i) Anusaarka: Machine Translation in Stages

This system is based on the principles of Panini grammar and is used to translate the sentences from Punjabi, Kannada, Telugu, Marathi, Bengali to Hindi. This system consists of a pre-processor module, language analyzer module and processor-cum output reader. This system is tested on children's stories. In the first module the input text is pre-processed because of the Indian language influenced by local dialects. So the input text needs to pre-process. The next module called as core Anusaaraka, takes input from previous module and extracts all the information and part of speech tagging. It creates a pseudo code as output which is more closer to the target language. In the next module the domain specific knowledge is acquired and statistical processing is done on the output of previous module to get the text in target language. The last module is post processing which arranges the output text as per the target language format. The division of the work into two different sections like language analysis in one section and text generation in other gives early availability of the system, early

feedback to improve the next section from the first section, robustness, clear task division, interlingua evolution and makes it easier to develop the second section [36].

(ii) Punjabi-Hindi MTS

This system uses direct translation approach for translating the Punjabi text to Hindi Text initially. Architecture of this system has multiple modules. First module is text normalization module in which the input text is converted into unicode format because of internationally acceptable and simple structure of the Unicode. The next module is tokenization module, in which the input Punjabi text is splitted into individual sentences and each word is extracted from each sentence so that these could be mapped into corresponding Hindi words. Next is translation engine module in which Named Entity Recognition (NER) ambiguity resolution, mapping of words, repetitive construct handling submodules performs their corresponding tasks as per their names. This system uses various lexical resources like root word lexicon, inflectional form lexicon, ambiguous word lexicon, bigram and trigram table to perform various word to word mapping and ambiguity resolution during translation. The next module is handling module which handles the repetitive constructs found in translation process. The last module is the target language (TL) generation module in which some rule base is used to map the source language (SL) grammatical structure to the TL grammatical structure. The accuracy of this system has been reported 90.67% and the intangibility of the output text has been reported 2.72 out of 3 with word error rate of 2.34% [37].

(iii) Hindi to Punjabi Web Based MTS

A web interface based MT system is proposed for the two syntactically close Indian languages — Hindi to Punjabi and vice verse. It uses DMT approach for translation among two languages. Two dictionaries (Hindi-Punjabi) one from the BhashaVibhag and one from National Book Trust were digitized for preparing the database for the translation which consists of approximately one lakh words. The most important

module of this system was training the system with the existing database to generate the new entries. An automatic font converter is used to convert the non-Unicode input data font into the standardized Unicode font for better processing of the sentence. The system also handles the word sense disambiguation by using two database files, one with words of no dis-ambiguity and the second one having multiple meaning with the context of the word in which it is used. Accuracy of this system was found to be 95%. The system is available online at free of cost and could be used in many applications [38].

(iv) Hindi-Dogri MTS

This system uses direct approach of translation for translating Hindi text into Dogri text. This system has a pre-processing phase. In this phase normalization of input text and the identification of proper noun and the words which can not be translated word to word is done. The next phase is the target text generation which performs the tasks of lexical look-up, handling of "Kar", disambiguation and analysis of the inflected phrases. The output generator phase is used for final generation of the target language [39].

(v) Shahmukhi to Gurmukhi Transliteration System

This system removes the scripting barriers between Pakistani Punjabi people which are using Shahmukhi script for writing the Punjabi Texts (Old Punjabi Historic Texts) and the Indian Punjabi people which are using the Gurmukhi script for writing the Punjabi texts (Modern Punjabi texts) to share with each other. The system have two parts for transliteration, first is pre-processing part in which Shahmukhi script is entered in the Unicode format and is tokenized with the help of dictionary look-up process. If no exact match occurs then the rule base approach is used to find the best possible equivalent Gurmukhi token. The second part of the system is post-processing in which the contest analysis of the Gurmukhi tokens is performed by aligning Unicode of the

tokens. The missing diacritical marks are handled by all form generator module of the second part and the queue manager module of post-processing part works on the Bi-gram model [40].

(vi) Telugu-English MTS

This system has five modules and the first module is pre-processing module which converts input Telugu sentence into ROMAN script format. The second module is the morphological analyzer which performs morphological analysis of the input text i.e. part of speech tagging, chunking etc. The third module is translation module which translates Telugu words to English words using a Telugu to English dictionary of more than 2000 word. The fourth module is word order maintenance in which the word order of the output is maintained as per the target language. At last the reverse morphology is applied to the output English text. The system was tested on a set of 411 Telugu sentences. The system was compared with the web based Google translator and reported to be 60% better than the google translator [41, 42].

2.3.2 Rule Based Machine Translation (RBMT) systems

These systems are divided into two parts as Interlingua Based Machine Translation (IBMT) and Transfer Based Machine Translation (TBMT) Systems.

IBMT systems

(i) Interlingua Machine Translation: a parametric approach

UNITRAN systems is parametric based Interlingua machine translation system which translates English, Spanish and German languages bidirectionally. This system has used the Interlingua approach to store the language-independent part and the language-specific part is represented by the parameter settings. The proposed system has been compared with direct translation and transfer based approach and found that the current approach helps in cross language translation using the common Interlingua

representation. The proposed system comprises of two level of processing, one at syntactic level and other at lexical semantic level. The system uses 20 parameters and 150 items for the lexical semantic analysis per language. The scope of the system is single isolated sentences [43].

(ii) ANGLABHARTI

The system uses context free grammar with pattern directed approach over the rule base extraction for analyzing the source English Language sentence. After analyzing the sentences system generates a pseudo-code structure. The design of the system contains a corpus of patterns followed by rule acquisition system module which acquires the set of rules from corpus analysis to identify possible component. Sense disambiguator module is used to remove all the possible ambiguities in the source language. Pattern directed parsing module is used to generate the pseudo-interlingua code with the help of multilingual lexical database, online lexicon and sense disambiguator module. The target language generator module is used to generate the target language from the pseudo target-interlingua code for corresponding target language. A corrector module is used to handle the ill formed sentences followed by the post editor module. As per the Vanquish triangle of machine translation taxonomy, this system lies between the transfer and Interlingua approach because the translation is used for a group of target language but it ignores the complete understanding of the source sentences [44].

(iii) AnglaHindi MTS

This system is derived from the ANGLABHARTI system for translating English to Indian languages and uses a pseudo-interlingua approach for translation. The English text is analyzed once and an intermediate pseudo code is generated which contains most of the word-group and word of the target languages. The ANGLABHARTI approach shows that only with 30% effort a new translation system could be easily generated from the intermediate pseudo code. ANGLAHINDI system uses all the

modules of the ANGLABHARTI system with addition of abstracted example base for translating the noun and verb phrases. This system first calls the example base of ANUBHARTI system and then calls the rule base module of ANGLABHARTI system. The ambiguities in the example base are resolved using distance function. The system accepts text of any format that may be quoted text, currencies etc. The system performance is evaluated using human evaluation system and for simple sentences with maximum 20 words length, gives 90% good translation. The system has no particular domain but could be used for any specific domain with the domain specific example base [45].

(iv) English-Hindi Interlingua Based MTS

The proposed MTS translates English sentences into Hindi sentences using interlingua approach. The interlingua taken in this systems is UNL. The system uses Language–UWWord dictionary which stores the language dependent as well as language independent attributes. This system have three modules as Hindi Analyzer to convert Hindi text into UNL expressions, English analyzer to convert English Text into UNL expressions and the Hindi Generator which converts the UNL expressions into Hindi text. The analyzer module uses the rule base to map English and Hindi words to UW's and UW's to Hindi words, transforming the semantic and syntactic dependency in Hindi and English language into corresponding UNL attributes and relations. The generator module uses the Language–Universal Word dictionary and the set of generator rules to generate the Hindi text from the UNL format. The system was tested on a set of 180 sentences given by United Nations University and obtained an accuracy of 80%. The scope of this system is not limited to techno-scientific domain but also works on literature as well [46].

(v) Tamil to UNL Enconverter

The Tamil Enconverter system converts Tamil sentences into UNL expressions which

are language independent. This system uses a UNL Knowledge Base (KB) to store all possible UNL relations and a database of enconversion rules to map the Tamil words with their source language dependent relations to their corresponding UNL words and UNL relations. A Tamil-UNL word dictionary is also used in this system. The necessary information required from the source language for translation purpose is obtained by morphological analyzer and syntactic functional grouping at syntax level. It uses this information to map a word of Tamil to UNL with the help of Tamil-UNL dictionary and rule database. A node list is prepared initially and then modified as per the corresponding rule used during conversion process [47].

(vi) UNL Deconverter for Tamil

This system is used to translate the UNL data/ expressions into Tamil language using the interlingua approach where the UNL is itself an interlingua. The first module of this system is semantic module in which the input sentences are classified and their types are identified. Then the root verbs are generated with the tense, mood, aspect, intention attributes of the verbs and the subject and verb agreement is also checked here. The output of this module is a network whose nodes are the grammatical phrases like Noun, Verb, Adjective, Adverb etc. This output is sent to syntactic module which identifies the position of various grammatical components into the sentence. After finalizing the syntactic structure the next module performs the morphological analysis for the target language and also inserts the verb endings [48].

(vii) ANGLABHARTI-II Architecture

This system is an enhancement of the ANGLABHARTI machine translation system with an addition of computer assisted tools to real life sentence translation from English to Indian Languages. The new modules which are added to the ANGLABHARTI system are pre-editing module, multi-word expression module, raw and generalized example base module, automated pre-editing module, failure analysis module, learning

user's lexical choice module and ill-formed sentence corrector module [49]).

(viii) Bengali Case Structure using UNL Constructs

In this system the input Bengali sentence is first converted into UNL expressions known as EnCo and then from UNL expressions to the Bengali language known as DeCo to do the analysis and generation of the Bengali language case structure constructs. The UNL is taken as the Interlingua for the verification of translated source text. This system uses a set of analysis rules for enconversion process and the set of generation rules for deconversion process. The case structure is identified by the Karakas analysis in Bengali. In Bengali there are six types of Karakas, so the system uses the six types of Karakas with the set of rules for the EnCo and DeCo for the Bengali Language [50].

(ix) Bi-lingual Hinglish MTS

This system is a mixture of the ANGLABHARTI-II and ANUBHARTI-II systems with an additional layer. This system is used to handle the mixed Hindi and English text and converts the mixed text to pure Hindi and pure English text. The Roman script is used for writing the mixed text. The system is capable of handling the complex code mixed (CCM) text i.e. at the clause level as well as simple code mixed (SCM) text i.e. at word level. The translation is done in seven steps. In the first step, the mixed text is morphologically analyzed by the Hindi Morphological analyzer and the part of speech tagging is done for the Hindi recognized words and the unrecognized Hindi words are marked. Some of the words from the sentence may be recognized as Hindi words although they are English words due the Roman script. In the second step, the mixed sentence is analyzed with the English analyzer and the same is repeated as in the step one. In the third step, the unrecognized words from step one and step two are analyzed for the plural form of the Hindi text as from the English morphological analysis and same for the English text. The words which are still unrecognized are treated as the

names. In the fourth step, the complex code text is divided into two parts as simple code mix text for Hindi and simple code mix text for English. In the fifth step, the simple code mix of Hindi text is translated into pure Hindi text and in the sixth step, the simple code mix of English text is translated into pseudo English code and then it translates this to the pure Hindi text. In the seventh step, if target language is Hindi then the output is obtained into the step six and for English the system needs to translate the pure Hindi text [51].

(x) HinD

HinD is a Hindi Deconverter machine translation system for translating UNL expressions into Hindi sentence. This system has different phases for generating the Hindi text from UNL expressions. This system generates translations at single sentence level. The first phase is the UNL parsing and repair module in which the UNL expression is parsed in the form of a graph structure and with nodes as UW's and relations as arc and with UNL attributes attached to UWs. Errors if any in the UNL expression are also repaired in this module like UNL scope resolutions / insertions. The next module is lexeme selection in which the corresponding Hindi lexeme equivalent to each UNL lexeme is selected with the help of UNL-Hindi dictionary. Word sense disambiguation is not a problem here because that is taken care during Hindi to UNL enconversion process. The next phase is case identification and morphology generation in which the UNL relations are used to identify the Hindi case markers. A set of rules are used for Hindi Morphology generation. Function word insertion is the next module in which the insertion of case markers, conjunctions, relative pronouns is done and for this a set of rules are used. The next phase is the syntax planning in which the word order decision is done for the target Hindi language and for this several assumptions are made and according to them several rules is also proposed and finally the Hindi sentence were generated as an output. A set of 901 Hindi sentences are generated from the agriculture domain with BLEU score of 0.34 [52].

(xi) English to UNL (Interlingua) EnConversion

This system translates English sentences into Universal Networking Language expressions. This system uses a probabilistic parser to generate the phrase structure and the type dependency tree of the source language with the help of English language database. The Princeton wordnet has been used for the part of speech tagging, converting the English language dependent relations obtained from the dependency tree during parsing into their equivalent UNL relations and the semantic information about the source language text into corresponding universal word attributes. Phrase structure of the source language is used to position the corresponding UW into the UNL expressions. The system is divided into six phases; starting with parsing and ending with scope identification. The system is better in handling the compound concepts through scopes in UNL expressions. The system performance was evaluated on a set of 60 sentences taken from the agriculture field and reported a BLEU score of 0.26 [53].

(xii) English to Sanskrit MTS

This system is an extension of ANGLABHARTI–II architecture for translating the English Text into Sanskrit Text using the PLIL (Pseudo-Lingua for Indian Languages) intermediate code representation for the English sentences and a set of Panini Ashtadhyayi Sanskrit Grammar Rules. The input English sentence is first processed by ANGLABHARTI system and produces the PLIL code (Interlingua representation code). This PLIL code in the form of a parse tree is morpho-syntactically parsed using the Panini Ashtadhyayi rules. First, the semantic information is extracted from the nodes of the parse tree like the Karta, Karma, and Kriya etc. In the next step, the thematic roles (Karaka assignment and Lakara assignment) are assigned to the node arguments as well as to the root words. Finally the Pratyayas are attached to the morpho-syntactic words and the Sanskrit sentence structure is generated. The system uses the database of root words of the Sanskrit words for the equivalent English words. This system is working for the simple sentences [32].

(xiii) Punjabi to UNL Enconversion System

This systems translates Punjabi text into UNL expressions. The UNL project was started by United Nations University with an aim to share the knowledge over internet without any language barrier. Once the text gets converted into UNL (Enconverted) by using other language Deconverter, the text could be easily converted into the target language. The main focus of the system is of extracting the UNL attributes and relations from the Punjabi Language. This system uses the Punjabi shallow parser for POS tagging, tokenizing, morphological analysis. The system is divided into seven phases. The first phase is parsing using Punjabi Shallow parser, followed by a linked list of node creation having the output of the parser as Punjabi word, UW, POS, list of attributes. In the next phase, mapping of the source language i.e. Punjabi word to UW is done with the help of Punjabi-UW dictionary. The unknown words are identified in the next phase by looking in the case marker table. The system uses about one thousand Enconversion rules to apply on the linked list of nodes to generate UNL expression for the Punjabi text. The system uses 41 UNL relations out of available 46 and gives 95% of translation efficiency [28].

(xiv) Bangla Enconverter

This systems converts the Bangla sentence into into UNL expressions. The system uses the predicate preserving technique for the morphological, syntactic and semantic analysis of the source Bangla language text which identifies the main predicate in every iteration process. The system has a Bangla analyzer component which contains the rules for morphological analysis, composition analysis and resolving the dependent relations. The system uses the Bangla-UNL dictionary to map the Bangla word to their corresponding UNL words. The encoding of the Bangla sentence is done with the help of a shift-reduce parser. The system is tested successfully on the assertive sentences. The system is yet to be tested on the compound sentences [54].

(xv) Deconverter for Punjabi Language

This MTS translates UNL expressions into Punjabi sentences. The UNL is an interlingua approach of translation which requires only $2n$ components instead of $n(n-1)$ components in the traditional machine translation approaches. This system have five basic building blocks. The first is the UNL parser which creates the list of nodes in the form of directed acyclic graph. The next block is the lexeme selection in which the target language (Punjabi) root words with their corresponding characteristics are extracted from the Punjabi dictionary for each of the UNL word by looking into the UNL-Punjabi dictionary. After generating the target language words, morphological analyzer performs the morphological analysis in which the root form of the Punjabi words may be edited and more grammatical attributes may be added to the root word to remove any ambiguities with the help of a set of morphological rules. In the next block, the functional words are added into the target language sentence which are used as the case marker in the target language with the help of set of rules. The last block is syntactic linearization in which the target language structure i.e. the word order is generated for the output of the previous phase. This system is evaluated on a set of 1000 UNL–Punjabi sentences. The BLEU score of this system is 0.72. The system produces 89% correct translation of the test set and give faithfulness score of 92% and also has a 3.61 fluency score [55].

(xvi) UNL Deconverter Implementation for Malayalam

This system creates a framework for converting the UNL documents into the Malayalam language text. This system uses a morphological generator with the help of Malayalam word dictionary and set of rule to transfer the UNL text into the target language This system converts the UNL documents into the semantic net with entry node as the root of the net representing the main predicate in the text. Then the set of transfer rule (generating rules) are applied to each node in the Node-net to generate the corresponding words in the Malayalam language and the syntactic and semantic structure of the

target language is created by applying the corresponding syntactic and semantic rules on the word list obtained from the node-net. The system uses condition window for checking the generation window neighbor's condition for applying the generation rules and the generation window to check the adjacent nodes for applying the corresponding generation rule. This process continues till all the nodes in the Node-net become the part of the corresponding node list of the Malayalam. The system was tested on single UNL sentence level deconversion as well as on a corpus of 11 UNL sentences and it was found that the conversion to Malayalam Language is successful process. This system needs more generation rules and testing on more UNL corpus [56].

(xvii) English to Tamil using UNL

This system uses UNL as interlingua. At first it generates UNL code for English sentences (English Enconverter) and then from UNL code to Tamil text (Tamil Deconverter). The efficiency of this system over BLEU score was 0.581 and it outperforms Google translator[57].

TBMT systems

(i) Computer Assisted Translation System: An Indian perspective (MANTRA)

This system is a machine assisted translation tool used to decode the English text to Hindi with the domain of personnel administration. This system has pre-editing, parser, generator and post-editing modules. The input English text is pre-processed so that any spelling mistake could be rectified in the early stage and transform the input text in a form that system could easily generate the parse tree. The approach of this system for translation is lexical to lexical tree translation. In the next module, the system uses the VYAKARTA parser which was developed at CDAC Pune and uses the Tree Adjoining Grammar (TAG) formalism to generate the syntactic parse (lexical) tree to get the functional description of the input text. The output of this module is sent to the next generator module which generates the target Hindi text with

the help of English, Hindi and transfer lexicons. The grammar creation is done with the help of 'KOSHAKAR' which is a graphical system that guides the user in creating the grammar in an easy way. The tree acquisition module is used to make the entries in the parse tree for each word of the input text. This project was demonstrated to government of India with the specific domain of gazette notification on appointments in government of India after the approval from the government of India and its name becomes MANTRA-RAJBHASHA. This system is tested by various experts and gave 95% accuracy of translation in a specific domain. The latest version of this system is mantra-rajbhasha-ver.6.0 and easily down-loadable from the official website https://mantra-rajbhasha.rb-aai.in/. The scope of this system is extended to agriculture, banking, education, finance, health care, information technology and small scale industries [58].

(ii) VAASAANUBAADA

This system uses RBMT approach for converting the news texts from Bengali to Assamese. A bilingual database of Bengali to Assamese is prepared manually for this system. The system has pre-processing module, followed by generation module, inter-language matching and post-processing module. In the pre-processing module the initial cleaning and alignment of the bilingual example base (corpus) is done and then chunking of the input text is done using a particular partitioning algorithm into different chunks. A list of chunks is taken as output from this module. In the generation module the chunk list is matched partially with the example base of the SL as well as examples of the TL. If the input chunk does not match then backtracking is done and further segmentation of the input sentence is done using the break points and again the matching process starts. This process continues till all the words / chunks of the source language do not have a match in the example base. The word order management is not a problem in this language pair because of the common language structure. In the post processing module, numeric and the special characters which remain same in both the

languages are inserted into the target sentence at their correct position. The system is tested on a data base of 315 k and produces encouraging results [59].

(iii) English to Urdu RBMT

This system uses RBMT to do translation from English to Urdu Language. Set of hand crafted rules are used to transfer the SVO structure of English into the SOV structure of Urdu. A recursive swapping of the verb phrase in the source sentence parse tree generates the SOV parse structure for the Urdu. Context free grammar is used to find out the attributes in the structure of the language. To find out the tense, aspect, modality characteristics the system makes use of the Panini grammar approach in reverse and performs the translation. The system have four basic phases for the translation as lexical analysis, syntactic /semantic analysis, transformation and the generation phase [60].

(iv) GB Theory Based Hindi-English Machine Translation System (MTS)

This system performs the translation of Hindi sentence into English sentence using phrase rule based translation approach. The system uses the GB Theory for the making of the phrase rules. The system consists of various modules for translation. The parsing module performs the normalization of the input sentence followed by part of speech tagging using the source language lexicon and then the parser generates the parse tree corresponding to the tagged input text if the input text is grammatically correct according to the GB theory set of phrase rules. The system does not involve the semantic analysis during the parsing phase. The generation phrase performs the tasks of mapping the Hindi phrases from the parse tree to the corresponding English phrases. Finally by traversing the target language parse tree using in-order traversing approach the target sentence is generated [61].

(v) Sanskrit–Hindi Anusaarka: An Accessor-cum-Machine Translator

This system uses the enhanced version of the Anusaarka system for translating the Sanskrit text into Hindi. The new version of Anusaarka removes the problem of

training at the end and makes the system more user friendly and allows the linguistic resources developed to be pluged-in as and when required. The system takes the Sanskrit sentences from either a pdf, html, text file format and converts that into Apertium format. Using parser it performs the morphological analysis, part-of speech tagging and chunking. Using the C language intergrated production system as an expert system it translate the source text into target language [62].

(vi) Sanskrit Analysis System

Sanskrit analysis system is a analyzing system for the Sanskrit language with two basic modules as shallow parser and Karka analyzer. For the tokenization, the system uses the sandhi module which splits the complex words into small components. The shallow parser module is further divided into sub-modules of sandhi, samasa, subanta, genter, Kredanta, taddhita, tinanta analyzers and part-of-speech tagger module. The input sanskrit text in the form of Devnagri Unicode is processed by the shallow parser to parse the input sentences with tagging of various part of speech. The output of the shallow parser is processed by the Karka analyzer to find out the syntactic and semantic relations in the input sentence words. Some of the modules of this system are still under process and the integration of all the modules is also still under process [63].

(vii) Punjabi-English Noun Phrase RBMT

This system translates the noun phrases from Punjabi to English using the transfer based approach. The architecture of the system consists of pre-processing of the input sentence, part-of-speech tagging, word sense disambiguation, translation and the synthesis of the disambiguated tagged words in the TL. The system uses the analysis, translate, generation methodology for translating the Punjabi noun phrase into English. The accuracy of translation of this system comes out to be 85.33% on a new testing data set of 500 phrases [64].

(viii) English to Sanskrit MTS

This system uses RBMT approach to convert English text into Sanskrit sentences. The system has four modules. The first module is the lexical parser which takes input as English Sentence and parses the input sentence semantically with each word having its semantic role and grammatical relationships with other words in the sentence. The system uses a Sanskrit dictionary and a set of semantic rules to map the parsed English words to their equivalent Sanskrit words. The mapped Sanskrit words are then forwarded to the translator. Output of translator module goes to the composer module as input which takes care of the free word order structure of target language [65].

(ix) Transfer Based English to Sanskrit MTS

In this system, transfer based approach is used to translate the English sentences into the Sanskrit language sentences. In the analysis phase, the source language is analysed by tokenization of the input sentence, segmentation of the tokens using morphological analysis, phrase identification using the grammar of the English (source) language and parse tree generation. The second transfer phase, consists of transferring the source language parse tree phrase by phrase into the target language parse tree with the help of a bilingual dictionary. The third generation phase the TL sentence is generated from the target language parse tree with the set of TL grammar rules [66].

(x) Sanskrit Compound Processor

This system is used to process the compound word formation in the Sanskrit language. A compound word is the combination of more than one word and the meaning conveyed by that single word will be same as by those two or more words. The system has four modules. The first module is the segmentation in which the input compound word is divided into different segments which are morphologically valid words. This process of segmentation is done by applying the sandhi rules in reverse order. The output of the segmentation is multiple segments and to find out the valid segments the system uses the Optimality theory to rank the output segments and the higher

ranked segments are selected. Matching of the segments is done by using a Sanskrit corpus of sandhied and non-sandhied words. This output is taken by the next module of constituency parser which finds the exact way of binding the components together by using the binary tree to show the syntactic structure of the compound for each of the possible constituent component combinations. The next module is type identifier which inserts a tag in the parse tree based on the type of components involved in the compounds. There are 55 tags identified for this system. The last module is paraphrase generation which uses the tagged parser and the generator produces the paraphrase for the compound. The system is tested on a test data set of 400 compound words and reported 63% of precision [67].

(xi) Designing a Constraint Based Parser for Sanskrit

Parsing of sentence is the process of determining the grammatical structure underlying the sequence of words in the sentence. Statistical approach and grammar rule approach are the two ways of designing the parser for a language. Here the grammar rule based approach with certain constraints approach is used to design the parser for Sanskrit language. Mathematically the words in the sentence are represented in the form of a graph whose nodes indicates the words and the arcs represent the relationships between the words. Parsing of this graph is the process of extracting the sub graph from the main graph with directed tree characteristics. For designing such parser, four basic design principles are taken into consideration. The 5D matrix representation is used to represent the graph with all the words and the relations as $C\ [i, j, R, l, m]$. The performance of this system is evaluated on a test data set of 110 sentences taken from a school book. From the total dataset 86% of the sentences are parsed in the first parse [68].

(xii) Automatic Translating of Simple Sentences Punjabi to English

This system uses RBMT approach for translating the Punjabi simple sentences to

English from the legal domain. The first phase of translation is pre-processing phase in which the input sentences are restricted to only simple subject-object-verb format and all other complex and compound sentences are discarded manually. By using the data-set of joined words, the joining of two or more than two words is done automatically so that they give single equivalent English word in translation process. The next phase is tokenization which separates words from the sentence based on the blank space. The morphological analysis and POS tagging is the next phase in which a database of morphological information of each word is created with the help of POS tagger and set of rules are used to disambiguate the multiple tagging for a single word. The semantic information is obtained by adding Karka analysis in source language and this information is stored for the target language generation. In the next phase with the help of a bilingual Punjabi-English dictionary, word to word translation is done and the proper nouns are transliterated. The English language structure rules are used to synthesize the phrase as well as the sentence structure of the Punjabi sentence to transfer the grammar structure to Subject–Verb-Object format. Finally, the post-processing is done to remove any additional auxiliaries in the output sentence [69].

(xiii) Tagging Sanskrit Corpus using BIS POS Tagset

A new POS tagger for the Sanskrit language is proposed which is known as BIS POS. Although there are many POS tagger like JPOS, CPOS, IL-POSTS, ILMTPOS etc. The BIS POS tagger describes the morpho-syntactic nature of any language. The research on the Sanskrit language at syntactic level is still in its beginning phase and needs more work to be done. This tagger follows a hierarchical structure to do tagging of Indian language. At the level one (the outer most) there are eleven categories which annotates the linguistic features of Sanskrit language The top layer is further divided into two sub-layers. The nouns are categorized into four sub-categories. The pronouns are sub-categorized into five sub parts. The verb is divided into six sub-categories, two under the top sub-level one and four under the top sub-level two. There are three

sub-categories of the conjunctions and the conjunctions is the part of outer level one. The particle category in the tagset is divided into four sub categories. To identify the punctuation marks, unknown words, foreign words and echo words, the residual category in the tagset is used [70].

(xiv) EtranS

This software system performs the task of machine translation from English to Sanskrit language by adopting the RBMT approach. This system uses a sub set of the Context Free Grammar (CFG) known as synchronous CFG to do the linguistic representation of the language syntax. This system first uses the top to bottom approach followed by the bottom to top approach for the language translation model. Initially the input text is morphologically analyzed and POS tagging is done then with the set of rules a parser program checks whether the input sentence is grammatically correct or not. In the generation module, first the mapping on the source to target language is done using the set of transfer rules followed by the morphological analysis and finally a node to node translation of input sentence to target language is done. A set of five hundred sentences of three categories viz simple, large and extra-large are used to evaluate the system efficiency. The system reported 90 % of the successful translation of the source sentences to the target sentences [71].

(xv) English-Kannada UCSG based Machine Translation

This system uses the transfer based approach of machine translation to translate the English sentences into Kannada sentences with the domain of government circulars. This method makes use of the Universal Clause Structure Grammar (UCSG) format for the translation. The system takes English sentence as input and performs the analysis, parsing using the UCSG parser and the parsed sentence is then translated into Kannada with the set of translation rules and a bilingual English-Kannada dictionary. The target language sentence is generated using network based Kannada module. This system is

still in improvement stage [72].

(xvi) Sanskrit to English RBMTS

This system uses the rule based approach for translating the Sanskrit language sentence into English language sentence. The system uses three different algorithm for translation purposes. The first algorithm is used to do tokenization and extracting the root word and the suffix part from the input Sanskrit sentence. The next algorithm performs the task of syntactic and semantic analysis for the tokenized sentence. Third algorithm is used for the transformation to the target language with the help of transfer rules. The target language generator rules are used to generate the target language semantic and syntactic structure [73].

(xvii) English- Hindi RBMTS

This systems performs English-Hindi machine translation based on the dependency parsing approach where the domain restriction means the domain specific dictionary with the semantic attributes. The authors reduced the traditional analysis-transfer-generation model into analysis-generation model by combining the transfer-generation into single generation phase by adopting a descending transfer approach in the Vauquois triangle of machine translation approaches. The system parses the source language using a dependency parse along with new architecture and the syntax planning algorithm and then converts the parsed output directly in to the target language text. The system is evaluated using 1403 words in 100 sentences of English and 1231 words of Hindi language from agricultural domain. The current system performs better than the google translator [74].

(xviii) Sanskrit to English (TranSish) RBMTS

This system uses parser and semantic mapper for translating Sanskrit text to English text. It provides the text to speech facility and can do translation only for present tense [75].

(xix) English - Kannada/Telugu MTS

The system is used to translate the English text into Kannada/Telugu using the rule based and dictionary based hybrid approach. This system is also used to identify the source language of any text particularly for English, Kannada and Telugu. For training of this system, a text of 100 lines is used from each of three languages and using the identification algorithm the language identification is achieved. The translation process has four phases. Two bilingual dictionaries English-Kannada and English-Telugu are used for the dictionary based approach. In the first phase, the input English sentence is morphologically analyzed and tokenized into different tokens and a list of words termed as nodes with the morphological attributes is prepared. In the second phase, input sentence structure is matched with the sentence structures stored in the English-Kannada/Telugu sentence structure table. In the third phase, from the bilingual dictionaries the target language words are identified for each node in the list of nodes of input sentences. Using the rule base, word sense disambiguation and the target language sentence structure is generated. In the fourth phase, the final output is displayed [76].

(xx) A Deterministic Dependency parser with Dynamic Programming for Sanskrit

This system uses grammar approach instead of data driven approach because of non-availability of the large data set for Sanskrit language to apply data driven approaches. This system parses the input sentence from left to right and make a adjacency matrix to find the compatible relations using local and global constraints. The parser represents the sentence in the form of a graph with nodes as the words and the arc as the relations. It uses depth-first traversing algorithm for parsing of Sanskrit text and produces the parse tree. The system uses a shallow parser developed by Huet in 2007 to remove any unwanted combinations of solutions and to disambiguate the part of speech tagger ambiguity except the case marker ambiguity which still needs to be done. For the better understanding, this system is represented into three rows. The

first row represents individual word and their corresponding position. The second row represents morphological analysis for each of the word/node. The third row represents the relations/edge. For a sentence "e" of length of "n" words, this system produces (n-1) parses and the user has to select out of maximum n-1 parses for the correct output. The system is tested on a corpus of 1316 Sanskrit sentences and have applied 35 relations for the tagging and produces 63.1% of accuracy [77].

2.3.3 Corpus Based Machine Translation (CBMT) systems

Rule based MT systems are divided into two parts as Example Based Machine Translation (EBMT) and Statistical Machine Translation (SMT) Systems.

EBMT systems

(i) ANUBHARTI

This system uses a hybrid example based approach (HEBMT) for translating from Hindi to English Language. The hybrid approach combines the pattern based and example based approach. The example base is not in the raw format but it is an abstracted example base which reduces the size of the example base. This system takes Hindi sentence as input and after performing the morphological analysis it produces different syntactic units as output like noun phrase, verb phrase, adjective phrase etc. The particular portion of the abstracted example base is searched to match the input sentence with these syntactic attributes. If a match occurs then a matrix is formed of input sentence and particular portion of example base, then a minimum distance is calculated and sentence with the minimum distance is selected from the example base for the translation system. The distance function used in this system is a simple word to word matching function. This system uses finite state machine for the translation purpose. This system provides the generic base model that suits to translate any two Indian languages [78, 79, 80].

(ii) ANUBHARTI-II

This system is an extension of previous ANUBHARTI system. This system tries to overcome the drawbacks of the previous system by introducing generalized hierarchical example base with the rule base instead of simple abstracted example base. This system is proposed to be used to do translation from Hindi to different Indian Languages. This MTS uses shallow parser to parse the input Hindi sentence and after parsing, the parsed sentence is matched with raw example base first and then with generalized example base. If still no match occurs then the rule base module is invoked. The system has an automatic pre-editing module which is used to transform the input sentence into different segments so that the segments could easily be translated and again combined in the output if the complete input sentence does not match in the example base and a system failure occurs. This system also comprises of a statistical automatic post-editing module which take care of the syntactic structure and word order of the output sentence as per the target language. The system also has various other modules like Named Entity Recognition module, Error-Analysis Module, Domain Customization Module, Generalized and Conditional Multi word expression module [79, 81].

(iii) SHIVA Machine translation System

This system is developed by the IIIT Hyderabad and IISc Bangalore to translate the English sentence to Hindi. This system is based on the example based approach of machine translation [82, 83].

(iv) English-Hindi MTS by IBM

This system uses EBMT for translating English sentence to the Hindi sentence and is developed by IBM India research lab using a parallel corpus of English and Hindi language sentences. Later on the statistical based approach of machine translation replaced the example based approach. Various IBM language models are used and alignment of the sentence corpus is done and translation probabilities are calculated.

Finally the decoding module is used to get the final Hindi output. A Hindi-English machine translation system was also developed by IBM using the statistical machine translation approach. The language model score and translation score for the test data comes out to be 21.27 and 8.36 respectively [84].

(v) English to Sanskrit EBMT

This system uses example based approach for English to Sanskrit translation and describes the divergence of English and Sanskrit language. The system uses an example base which contains the morphological and functional information for the input English sentence and their equivalent Sanskrit translation and the root words. The ENGCG parser is used to generate the parse tree for the source English language. The equivalent Sanskrit parser is generated using the Gerard Huet parser taken from [85]. An online English-Sanskrit dictionary as a database is used to find out the Sanskrit equivalent English word [86].

SMT systems

(i) English-Hindi Statistical Machine Translation

This system is used to translate English sentences into Hindi using the SMT approach. There are three basic components of the SMT system as language model (LM), translation model (TM) and decoder. For any successful SMT, the most important criteria is parallel corpus design and the alignment of sentences. So this system uses a bilingual corpora of more than 500 sentences of English-Hindi from the freedom fighters history. The language model for this system is developed for target Hindi language using SRILM toolkit. The translation model is developed using Giza++ tool which identifies the probability of the source language sentence in the target language. Decoder performs task of maximizing probability for the generated sentences. Fluency score of the system was 2.693 and adequacy of the system was 2.93 for a test set of 90 sentences [87].

(ii) Google Translator

This system is developed by Franz-Josef in 2007 and is based on SMT approach for translating English sentences to other languages and vice versa. Currently it provides translation among 90 world languages. Gujarati, Hindi, Kannada, Tamil, Telugu, Malayalam, Marathi and Urdu are Indian languages among the 90 world languages [88].

(iii) Discriminative Machine Translation using Global Lexical Selection

(iv) Hindi to Punjabi MTS Evaluation

The strategy used for Hindi-Punjabi MTS evaluation is to first make the selection of set of sentences, then the testing of intelligibility and accuracy is done followed by Sentence Error Rate (SER) and Word Error Rate (WER). Evaluation is performed on the data using observation and the scoring is done accordingly. So in the first phase of evaluation, a set of sentences from daily news (10000), articles (3500), official language quotes (8595), blog (3300), literature (100450) is collected. For evaluating the intelligibility test a group of 50 people is selected and they give scoring in four categories as score 3 to score 0 with score 3 as clear and intelligible sentences and score 0 as unintelligible sentences. The evaluators gives 70.3% sentences the score 3. Only the sentences from the literature give less intelligibility of 87.4% than the other systems because of the different language dialects used by the writers. For evaluating the accuracy of the system a set of source text and the corresponding translated text is provided to the evaluators and again scoring is done in four levels. The accuracy of the overall system comes out to be 2.63. The WER for the system is found to be 5.2% and the SER is found to be 42.4%. So in comparison to Punjabi-Hindi of 92%, nCzech-Lithuanian of 69%, CESILKO of 90% and RUSLAN of 40% the accuracy of the Hindi-Punjabi machine translation System comes out to be 95.12% [89].

(v) An English-Hindi Statistical MTS

This system translates sentence from English to Hindi using SMT approach. This system has three components as translation model, language model, maximizing the probability for the generated sentences. This system uses three IBM Model 1,2,3 for the translation model with a bilingual corpus. An English-Hindi parallel bilingual corpus of 150000 sentences from different domains of government office document, news articles and magazines is used to train the translation model for this system. For the language model a database of 80 million Hindi words are used with the tri-gram language model to find the probability of a word in the target Hindi language. The dynamic programming decoding algorithm is used to maximize probability of the generated sentences. This system is tested on 1032 English sentences and produces a BLEU score of 0.1391 and NIST score of 4.6296 [90].

(vi) English to Urdu SMT

This system uses the Hierarchical Phrase Based model (HPBM) for translating English sentences into Urdu sentences. This system uses tree like structure to extract synchronous Context Free Grammar. The hierarchical translation reordering rules are used during the training of the system on the data set. But this system has a drawback of learning the large number of rules which will result in over generation output. This system uses the Enabling Minority Language Engineering (EMILLE) corpus for the training, translation and testing purposes. The training of the system is done using the GIZA++ toolkit and it uses the n-gram based language model using SRILM Toolkit. The testing of the system is performed using MERT toolkit. The BLEU score the system comes out to be 0.132 [91].

(vii) English–Urdu MTS

This system is based on SMT approach to do machine translation from English to Urdu. The system is used to translate the Sahih Bukhari and Sahih Muslim sentences into Urdu from English. The first step before the translation starts is to normalize the

text in the database and segment the database into different parts each for the training, tuning and testing. The system uses the Moses toolkit for the training purpose and IRSTLM for the language modeling. After the decoding phase the BLEU score of the system comes out to be 32.11% and after tuning this model on the DevSet, it gives the BLEU score of 37.10% [92].

(viii) Hindi to English Statistical Machine Translation

This system is based on SMT approach and uses two translation models as phrase based and hierarchical based as training models. The system uses Moses to train the translation models. As the name indicates in phrase based models, the text in both the source and the target language is divided into different phrases whereas in hierarchical, the phrases may be arranged in recursive way. Phrase alignment is done with the Giza++ tool. For training these language models SRILM tool is used. Minimum-error-rate training is achieved with the help of MERT tool. The top first output is considered as the default output. The system gives BLEU score of 21.18 and 21.10 for the two respective language models on the given test data set and considered these models as base-models. The system uses ILCI corpora from health domain and 25000 parallel sentences of Hindi-English language pair for experiment purposes as the test data set. The qualities of translation of these models are analyzed using feature vector and pre-trained regression model. The model which gives high regression value has good quality of translation than the other. The feature vector contains different number of feature sets and the BLEU score enhancement is obtained by combining the complete feature set with linear Kernel and gave 21.82 BLEU score. The system is compared with Google translator and Bing translator on the given test data set and the output shows that this system is far better than the Google and Bing [93].

(ix) The Hindi-English bi-directional Translation System of IIT Bombay

WMT 2014 is a platform to evaluate the different approaches of machine translation

by providing the standardized test data set and the large size corpus for English-Hindi languages. With the use of these resources, two systems are developed one for the Hindi-English using phrase based approach and other one is for English-Hindi language pair using the factored based approach with additional pre-processing and post-processing modules to remove the language divergence. First the English corpus is normalized using Stanford Tokenizer and Moses tool. The Hindi corpus is normalized by using the resources provided by WMT 2014. For training of the system 284832 English-Hindi sentences of 10-20 word length are used and for the testing 5000 parallel sentences of more than 50 word length are used. For translation of the English-Hindi the pre-order corpus prepared by rule base approach is fed into a Phrase Base SMT system. Next is the training of the factor based model by using super tag as a factor or the number, case as the factors. The BLEU score for the Phrase Based pre-ordered corpus with number and case a factors comes out to be 10.1 at the WMT14. For Hindi-English Translation a shallow parser is used to do chunking and a set of rules are used to do pre-ordering in the source side. The phrase based reordered Hindi-English produces BLEU score of 13.7 at WMT14 [94].

(x) Automatic Post Editing with SMT approach for Online Learning Techniques

This system is designed with SMT and online learning framework. This system uses the output of any of the generic machine translation systems like RBMT, SMT and EBMT etc. as an input to this system. This system is trained to correct the errors in the output of the generic MTS. This statistical APE system produces the output which is then corrected by the user and becomes the final output. The system uses the Europarl out-of-domain corpora for the training purposes. The system uses MERT and Moses statistical tools for the weight optimization and for labeling of corpus for the statistical modeling. The system uses the online web services like google, bing for online learning purposes. This system is assessed on different corpus EMEA, Xerox and i3media of different domains and for different language pairs. The generic machine translation

systems used for testing are RBMT, SMT and Web based systems. The system gives better BLEU score for the EMEA and Xerox corpus [95].

(xi) English to Urdu SMT

This system performs machine translation from English to Urdu using SMT approach. The system makes use of two SMT models in the implementation using the Moses toolkit. The phrase based (hierarchical) and the syntax based model. The system uses a parallel corpus of English-Urdu from various domains of technology, religion, news and culture containing of more than 79000 sentences. Out of this amount, 95% of the sentences are used to train the system and the rest are used for the development and testing of the systems. The system also uses a monolingual corpus of Urdu language. The system is tested on three official test sets and results shows that the hierarchical phrase based MT performs better than the lexical phrase based model [96].

(xii) An-English-Assamese Machine Translation System

This system uses SMT approach for translating English sentences into Assamese sentences. The system uses probability of the language model and translation model to find the probability of the TL sentence for the SL sentence. Bayes theorem is used to find Assamese sentence for an English sentence. The language model finds the probability of occurrence of a word after another word or after $n-1$ words known as n-gram language model. This model also helps in determining the word order sequence in the target language. A monolingual Assamese dictionary is used to do this type of analysis. The translation model uses a bilingual English-Assamese corpus to generate the Assamese sentence from the English sentence. In phrase based translation model the English sentences is fragmented into different phrases after doing the morphological analysis with the help of bilingual corpus and then statistical analysis is done on the different phrases. The third phase is the decoding phase which takes the output from the translation phase and selects the best probabilities of translation from the various

translation probabilities in the previous phase. It uses a heuristic search to find the best possible translation for the English-Assamese language pair. The system is tested with a bilingual corpus of 5000 sentences and gives satisfactory results [97].

2.3.4 Hybrid approach Based Machine Translation (HBMT) systems

(i) The EB-ANUBAD Translator: A Hybrid Scheme

This system uses a hybrid approach of rule based and transfer based machine translation to translate English sentences to Bangla sentences. The system uses pre-processing module which takes input a set of English sentences and finds the boundary for each sentence, separates each word from individual sentence. Then supply word by word to the morphological parser which uses a bilingual English-Bangla electronic dictionary and a suffix file. The output of this module will be the root word and all the part of speech tagging information. This information is given to the rule based POS tagger which may give multiple possible tags to a single word and this ambiguity is resolved using the ontological information in the ontology analyzer module using 2-4 gram scanning and at last corresponding Bangla word is generated. This process is repeated for all words for a sentence. Once the tagging of all the words in a sentence is completed the corresponding Bangla sentence is generated using transfer approach with the help of Bangla grammar rules [98].

(ii) Matra

This system is a fully automatic MT with no human intervention to convert text from ENGLISH to Hindi with the general purpose domain and produces the indicative translation not the perfect translation. This system uses a hybrid approach of SMT and RBMT for the translation purpose. It uses statistical approach for the parsing, word sense disambiguation, abbreviation resolution and transliteration for the unknown words. The rule based approach is used in the lexical and structural transformation of the English text to the Hindi text. For the structural transformation the system uses an in-

termediate representation known as Matra Simple Intermediate Representation (MSIR) and stores all the syntactic as well as semantic knowledge in this representation. The system uses various modules for the translation. The first module is the pre-processing module which performs the task of splitting the input English text into individual sentences. Abbreviation resolution, numeric expression, address sections and acronyms are identified in this module. The next module is POS tagging and chunking in which the role of each word is identified in the sentence and their grammatical attributes are attached with them like gender, number, person, tense, aspect, modal etc. and all this is done using fnTBL tool. The next module is word sense disambiguation in which the mapping of English word and phrases to their corresponding Hindi word and phrases is done. The next module is the Sentence structuring module in which the MSIR format is generated for the output of the last module. The next module is the Target language generation module in which the TL is generated from the MSIR representation and this is done using the rule based generation engine. The last module is the transliteration module in which the unknown words in the sentence are transliterated into Hindi word using genetic algorithm and a pronunciation dictionary. The performance of the system is evaluated on a test dataset of 315 sentences taken from news archives and other resources. The BLEU score of Matra1 is 0.0377 and that of Matra2 is 0.0534. The system produces more than 65% translations as acceptable output [99].

(iii) Machine Translation: The Shakti Approach

This system is used to translate the text from English to Indian languages using hybrid approach of RBMT and SMT. The system is based on the language analysis phase, transfer phase and target generation phase. The architecture of this system is divided into 69 modules which are further divided into three phases. Currently this system is available for three languages Shakti Hindi, Shakti Marathi and Shakti Telugu out of which Shakti Marathi is not operational. The URL of the System is http://shakti.iiit.ac.in/ [79, 82, 83].

(iv) SAMPARK

This system is developed by the combined effort of 11 groups under a common roof named as Indian language to Indian Language MTS (IL-MT). Under this system, 18 Indian language pair machine translation is done and these are between Hindi <-> Urdu, Punjabi, Telugu, Bengali, Tamil, Marathi, Kannada, and between Tamil <-> Malayalam, Telugu. This system uses a hybrid approach of rule based, dictionary based and statistical based. The architecture of this system is based on a black board architecture because this system consists of a set of heterogeneous modules which are operating on a common stored data structure and could work in any sequence. Failure of one module does not stop the working of the whole system and new modules could be added easily. The system uses the Shakti standard format for input and output to each module. This system is absolutely automated system and does not need any human intervention at any stage of the translation [100].

(v) A Punjabi to Hindi Transliteration System

This system transliterates Punjabi text in the form of proper names of person, places, and foreign names to the corresponding Hindi text. This system uses three phases for the transliteration process. The first phase is the letter to letter mapping. Gurmukhi and Devanagari are the scripts for Punjabi and Hindi language respectively and are having the phonetic sound of the corresponding language letters which are used to find out the relations between the letters of these two languages. Because of the similarity of these two scripts, letter to letter mapping from Punjabi to Hindi becomes easy and this is done in this phase with the input taken in unicode format. But letter to letter mapping does not cover all letters of source language and this problem is handled in the next phase. In this phase, some hand crafted contextual rules are used to handle problems in previous phase. A total of 11 rules are crafted for the source language and 8 rules for the target language with the help of a linguist. These rules are then applied on the corresponding languages. With the application of these rules there is

an improvement in transliteration process but still needs more rules to cover up all aspects of the languages. Still some letters of the target language remains unresolved. This Soundex code scheme is applied to the Punjabi to Hindi transliteration system to handle the anomalies of the previous phase. A table of Soundex code is prepared to map Hindi compound characters to Punjabi equivalents. With the application of Soundex approach the efficiency of transliteration of the proposed system has increased. This system is tested with the benchmark sampling method on a set of 3500 names of people, 1500 name of different locations and 1000 words from other languages. The efficiency of the first phase comes out to be 73.13%. The application of rule based followed by Soundex phase gives 92.65% efficiency [101].

(vi) English to Sanskrit RBMT and ANN based MTS

In this system Artificial Neural Network (ANN) approach is used in combination with the rule based approach to do the translation from English to Sanskrit Language. The feed-forward architecture of the ANN is used to find out the equivalent words in Sanskrit language for each word of the input English sentence. The ANN module have three parts to do matching process. The first part is encoding of user data vector (UDV), a data structure to represent the input and output words. In encoding scheme, the input English words are coded into decimal format by dividing each alphabet by 32 and then converted into decimal form. The system uses separate UDV for the English noun, verb and Sanskrit noun, verb forms. The second phase is the input-output generation which uses the five character representation of the English UDV and Sanskrit UDV. The third phase is the decoding which is reverse of encoding scheme. The architecture of the system works in different modules. The first module is sentence tokenizer module in which the input English sentence is splitted into different tokens and provided to the next part-of-speech tagger module which uses hand crafted set of rule to identify the grammatical information like noun, verb, adjective, adverbs, pronouns etc. for the input tokens. The next Module is rule base engine which consists of set of rules to find out the

subject/ kartaa, root form of the verb/dhatu, object, adjective, prepositions etc. using the ANN part of the system for each of the English word in the Sanskrit language. The system then uses root dhatu extraction module, word extraction module, dhatu form generation module, word form generation module, sentence generation module by concatenation of kartaa, adjectives, karmaa, adverbs and verbs which gives the output Sanskrit sentence. The system is evaluated on different types of 20 English sentences and gives satisfactory performance [102].

(vii) English to Urdu ANN based MTS

This system uses translation rules and feed forward back propagation architecture of ANN to do the translation between English and Urdu language. The ANN system works as knowledge base for bilingual dictionary and the linguistic rules. The main part of the translation is the transformation of SVO English structure into the SOV Urdu structure and this is achieved with the help of the set of translation rules. The source English sentence is first analyzed with the help of a parser and tagger. The source language grammar structure is identified using ANN then the source language grammar structure is mapped with the target language grammar structure. After mapping the structure, ANN is used to map source language words with the target language word using bilingual English-Urdu dictionary and then with the help of linguistic rules the target language text is generated. The BLEU score for n-gram language model were found to be 0.6954, METEOR 0.8583 and F-score of 0.8650 [103].

(viii) Hindi-Punjabi MTS

This system translates two closely related Indian languages i.e. Hindi and Punjabi by adopting a hybrid approach of both direct and rule based machine translation approaches. The system architecture comprises of three components as pre-processing phase, translation Engine and post processing. In pre-processing phase, the system performs text normalization, replacing collocations and proper nouns followed by

generating tokens as meaningfull words from previous stages. In translation engine phase, various activities like identification of surnames, titles, ambiguity resolution and handling of unknown words are performed. The system is using Punjabi unigram database which is containing about 2,00,000 Punjabi words for identifying the correct words. The post processing phase of system performs the task of grammar correction and generates the target text as output. The accuracy of the system is 87.60 % and in comparison to other systems the performance of this system is far better [104].

(ix) Quantum Neural Network based MTS for Hindi to English

This system uses a fusion approach of rule based and quantum neural network (QNN) based to do translation of Hindi sentence into English sentence. The input Hindi sentence is first processed by the rule based system to distinguish the sentence types and tokenization purpose. The output of the rule based is inserted into the QNN architecture which performs the task of part of speech tagging and training the system using test data set of 2600 Hindi-English sentence pair. Three digit numeric codes are used to do the part of speech tagging for the words. Once the system gains the syntactic and semantic knowledge using the QNN then 500 tests are performed to check the performance of the system and compared with Google and Bing translator. The score on BLEU scale of the system was 0.7502 [105].

2.3.5 Neural Machine Translation (NMT) systems

Machine translation among eleven Indian languages using NMT approach have been proposed and has obtained better results than the traditional SMT approach based systems [35]. Microsoft provided NMT based translation support for 21 languages and added Hindi recently [106].Wu et al. [107] also uses NMT approach over the existing SMT approach and shows better results than SMT. Facebook in 2017 proposed the implementation of NMT using convolutional neural networks and claimed faster performance [108, 109]. Amazon has also launched its machine translation system using NMT approach [110]. Some important

platforms useful for the development of NMT systems includes Tensorflow, Torch, Theano, PyTorch, Matlab, DyNet-lamtram and EUREKA [111]. Neural network processes only numeric form of data. So to process text data through neural network first it has to be encoded into numeric form before processing further. The encoding could be done at character level, word level as well as at sentence level. Number of researchers have proposed several encoding schemes [112]. One-hot encoding [113] is a basic encoding scheme for representing words. Word2vec embedding [114, 115] has two models for embedding the words that are continuous bag of words (CBOW) and skip-gram models. Both the models have used a particular window size to predict target word from context words or context words from target word. Glove embedding [116] has also used global context in comparison to word2vec which was using only local window context. FastText embedding has [117] used CBOW for text categorization. FastText technique has used sub-categorized word n-gram information for the semantic relation identification among characters of word. Embedding from language models (ELMo) has [118] used two-way language models (forward as well as backward LSTM) for embedding the text in to numbers. Open artificial intelligence- generative pre-training (OAI-GPT) proposed by [119] has been used to find the semantics of words in application context domain. It has used one-way language model with transformer to extract semantic features from words. Bidirectional Encoder Representations from Transformers (BERT) proposed by [120] has used bi-transformer technique to extract semantic knowledge from the sentences. For the purpose of part-of-speech tagging, character based encoding has performed significant role [121, 122]. Character based encoding have been used in several applications including POS tagging [123, 124], morphological analysis [125], parsing [126], language modeling [127] in the field of NLP [128, 129]. The overview of various MTS has been shown in Figure 2.10

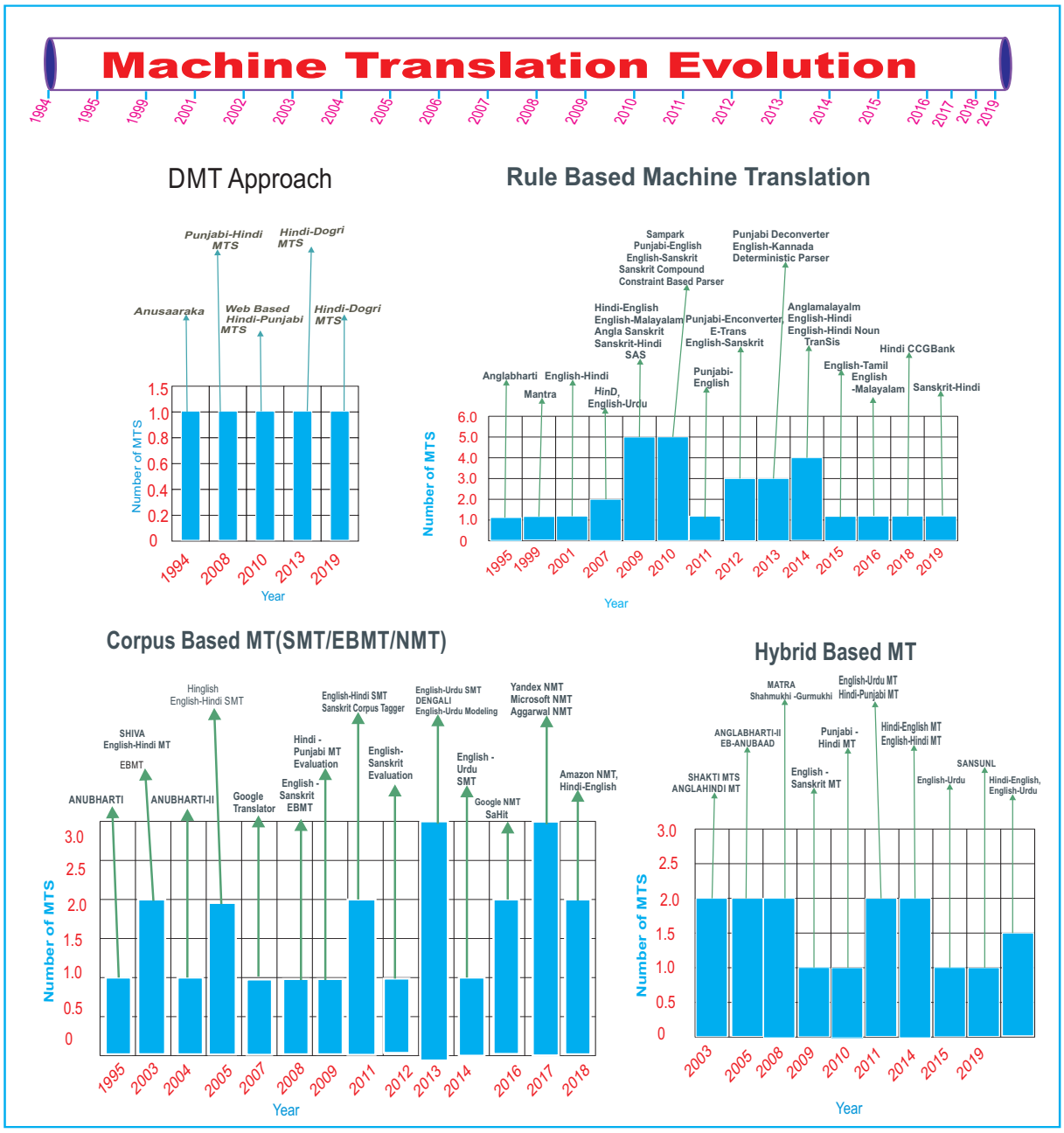

Figure 2.10: Evolution of MT in Indian perspective based on Different Approaches

2.4 Machine Translation Platforms and tools

This section gives an overview of some statistical tools, parser and corpus which are available online for developing new MTS. Table 2.1 shows some of the popular MTS platforms which could be used for developing new MTS. Various language corpora available for Indian

languages are also highlighted. Enabling Minority Language Engineering (EMILE) contains three types of corpora : parallel, monolingual and annotated. In Parallel corpus it contains two lakhs words for Bengali, Gujarati, Hindi, Punjabi and Urdu to English and visa-versa. Twenty annotated Hindi files are there in the corpus.

Gyan Nidhi corpus contains fifty thousand number of pages as a parallel corpus for each of eleven Indian languages (Assamese, Bengali, Gujarati, Hindi,Kannada, Malayalam, Marathi, Oriya, Punjabi, Telugu, Tamil) and English language also.

Open Source Parallel Corpus (OPUS) contains parallel corpus for Assamese, Bengali, Bhojpuri, English, Gujarati, Hindi, Kannada, Kashmiri, Konkani, Malayalam, Marathi, Oriya, Punjabi, Sanskrit, Tamil, Telugu and Urdu.

ILCI (Indian Language Corpora Initiative) contains a corpus of 50000 parallel aligned sentences in Bangla, English, Hindi, Gujarati, Konkani, Malayalam, Marathi, Oriya, Punjabi, Urdu, Tamil and Telugu in the domain of tourism and health.

Table 2.1: Popular MTS Platform

MT Platform	Language Pair	Domain	Features	Organization	Citation
Apertium Platform Open Source	Multilingual	General	Language Independent	Apertium	[130]
OpenNMT	Language Independent Multilingual	General	Dependency free, simple, compatible to any language pair	Systran, harvard NLP	[131]
Stanford NMT	Multilingual	General	BLEU score of 5.2	Stanford University	[132]
Microsoft Translator Hub	Multilingual	General	Supports 60 language systems and 10 speech systems, produces netter results	Microsoft	[133]
Google Translator	Multilingual	General	60 % reduction in error of translation using GNMT	Google 2016	[107]
Yandex Translator	Multilingual	General	More fluent and human like translation	Yandex	[134]

Table 2.2: Online Resources

Resource	Citation / URL
MTS	
Moses Statistical MTS	[135]
Cunei Hybrid for Example Based and Statistical MTS	[136]
Joshua Statistical MTS	[137]
Language Modeling Tool	
Neural Network Joint Model	[138]
IRSTLM Toolkit open source	[139]
CMU-Cambridge Statistical Language Modeling Toolkit v2(Open Source)	[140]
SRILM ToolKit (Open Source) 7	[141]
Neural Probabilistic Language Model Toolkit	[142]
Shallow Parser	
For Bengali,Hindi, Kannada, Malayalam, Marathi Punjabi, Tamil, Telugu	[143]
Complete Parser	
Malt Parser (language Independent)	[144]
For Hindi, Tamil, Telugu, Urdu	[145]
Parallel Corpora	
Gyan Nidhi	[145]
EMILLE	[146]
ILCI	[147]
OPUS	[148]
Golden Standard Dataset for Validation of UNL based translation	
Spnaish Server	http://www.unl.fi.upm.es/english/fr_examples.htm
UNL Web dataset	http://www.unlweb.net/wiki/Corpus
UNL-NL Dataset	
Gerard Huet Sanskrit-UNL dataset	https://gitlab.inria.fr/huet/Heritage_Resources
Hindi-UW dictionary	http://www.cfilt.iitb.ac.in/~hdict/webinterface_user/

2.5 Research gaps

Although, lot of work has been done in the last three decades for developing MTS with different language pairs (Indian languages) and for various domains. With the emergence of NMT approach, easy availability of high computing resources and corpus for Indian languages has created several new opportunities for the researchers to work in this field. The researchers are now more focused to apply machine learning algorithms for text processing rather than other fields and as a result several new tools and platforms are available for text processing. It is very difficult and time consuming process to create the rule base which will cover all aspects of language specifically for Sanskrit and Hindi languages which are highly inflected and morphological rich in nature. To apply SMT approach, the need of large corpus is again a big hurdle for language like Sanskrit. The following are some of the research avenues with which the researchers can start their research work:

- Although the ANGLABHARTI system uses Interlingua approach for translation from English to Indian Languages, but the system was not 100 % automatic and the Interlingua was again dependent on the target language for translation.
- In ANUSAARAKA system, the output produced was not grammatically correct and the user needs extra training for understanding the output because output follows grammar of the source language.
- The application area of the MANTRA system was restricted to official's domain and the grammar developed was capable to accept, analyze and generates sentence constructions in the office domain only.
- In VAASAANUBAADA machine translation, lot of preprocessing and post processing has to be done for translation. If there is any spelling mistake either in input or in the corpus then this will give the mismatch.
- In ANGLABHARTI-II, for the removal of deficiencies of RBMT different modules

have been introduced but the efficiency was only 80 %.

- In MATRA, the handling of interrogative, subjunctive, imperative moods and phrasal verbs needs to be handled properly.
- In ANUBHARTI-II, the generalization of example base is dependent on target language. Human post editing is performed to introduce determiners that are either not present or difficult to estimate in Hindi.
- SHAKTI machine translation system is capable of translating English language to only three Indian Languages.
- In UNL based English-Hindi machine translation, the language to Universal Word dictionary has to be enriched both in terms of universal word content and semantic attributes so as to capture the word and the world knowledge. The analyzers need to be augmented with powerful word-sense disambiguation modules.
- The ANUBAAD Hybrid Machine Translation System is a domain specific MT system limited to only news domain.
- The Hinglish MT System is unable to resolve the meaning in case of polysemous verbs due to a very shallow grammatical analysis used in the process.
- The ANUVAADAK MT System is again not general purpose and it is having the domain of office use only.
- In EBMT systems (English to (Hindi, Kannada, and Tamil) and Kannada to Tamil language pair) the manual preparation of parallel example base is a very challenging task.
- Google Translator uses statistical matches for translation and the translated text can often resulted into nonsensibal and obvious errors. Even in some cases inverting sentence meaning has also happened.

- The Sampark MT System was not easy to use and not interactive as well.
- The Hindi deconverter system needs to be plugged in an Interlingua based MT System with Hindi as the target language. The main challenge was the naturalness of the output and high fluency scores.
- Need to develop a platform like Snowball for creating the rule base in an easy and fast manner.
- Need to Create small modules which can enhance the performance or reduce the response time of the existing MTS like Named Entity Recognition (NER) tool, automatic pre or post processing tools using machine learning techniques.
- For MTS using UNL as interlingua approach, the resolution of UNL relation is a challenging area because it requires thousands of rules to resolve all the 55 plus UNL relations. So machine learning approaches can be used over the UNL dictionary to predict the possible relations with the Case marker module.
- Need to develop POS tagger or stemmer for Sanskrit and Hindi languages using hybrid approach of rule base and machine learning techniques.
- Need to develop automatic Kaarka Analyzer (case marker) for Sanskrit and Hindi by making use of the similarity features among Indian languages in such a way that only small effort is required to make this system for other Indian languages.
- Anaphora or Catphora resolution is still a challenging task for Sanskrit language. So special modules need to be developed for such types of problems which can be easily merged with the MTS adopting modular approach.
- Need to develop Sanskrit EnConverter and Deconverter system using UNL.
- Need to Develop new Operating Systems for computers using less ambiguous language like Sanskrit.

- Need to develop tools to extract text from scanned images and develop automatic digital corpus for languages like Sanskrit and Punjabi.

2.6 Problem Formulation

Based on the research gaps and the trends of the machine translation, the focus of this work will be on developing a machine translation system for Sanskrit to Universal Networking Language.

In this thesis, it is proposed to develop an EnConverter for Sanskrit Language to UNL, this will make us capable of translating Sanskrit to other approximately 25 languages.

2.7 Objectives

The main aim of this research is to design and develop a machine translation system for Sanskrit language to Universal Networking Language expressions. To achieve this goal following are the objectives:

(i) To study the Sanskrit (source) language for the knowledge of language structure.

(ii) To propose an enConverter for Sanskrit to UNL likely using Government and Binding (G.B.) Theory (Minimalistic Approach).

(iii) To implement the proposed enConverter for Sanskrit to UNL.

(iv) To test and demonstrate the use of the whole system.

Chapter 3

Proposed Sanskrit to UNL Enconversion System : SANSUNL

This chapter presents the proposed layered architecture of Sanskrit to UNL MT system. The proposed architecture consists of seven layers such as Pre-processing and Tokenization layer, POS tagging layer, Parsing layer, Node-list creation layer, Case marker identification layer, Unknown token handling layer and UNL generation layer. The functionality of each layer is explained in different sections of this chapter.

3.1 Proposed Sanskrit to UNL MT architecture

Sanskrit EnConversion is a process of converting Sanskrit text into UNL expressions. The objective of this research work is to develop Sanskrit language EnConversion to UNL system. The layered architecture of the proposed system known as "SANSUNL" is designed and shown in Figure 3.1. Each layer of the proposed architecture performs a specific task for processing the input text and generating the UNL expressions. The detailed discussion of each layer of the proposed architecture is given in next sections.

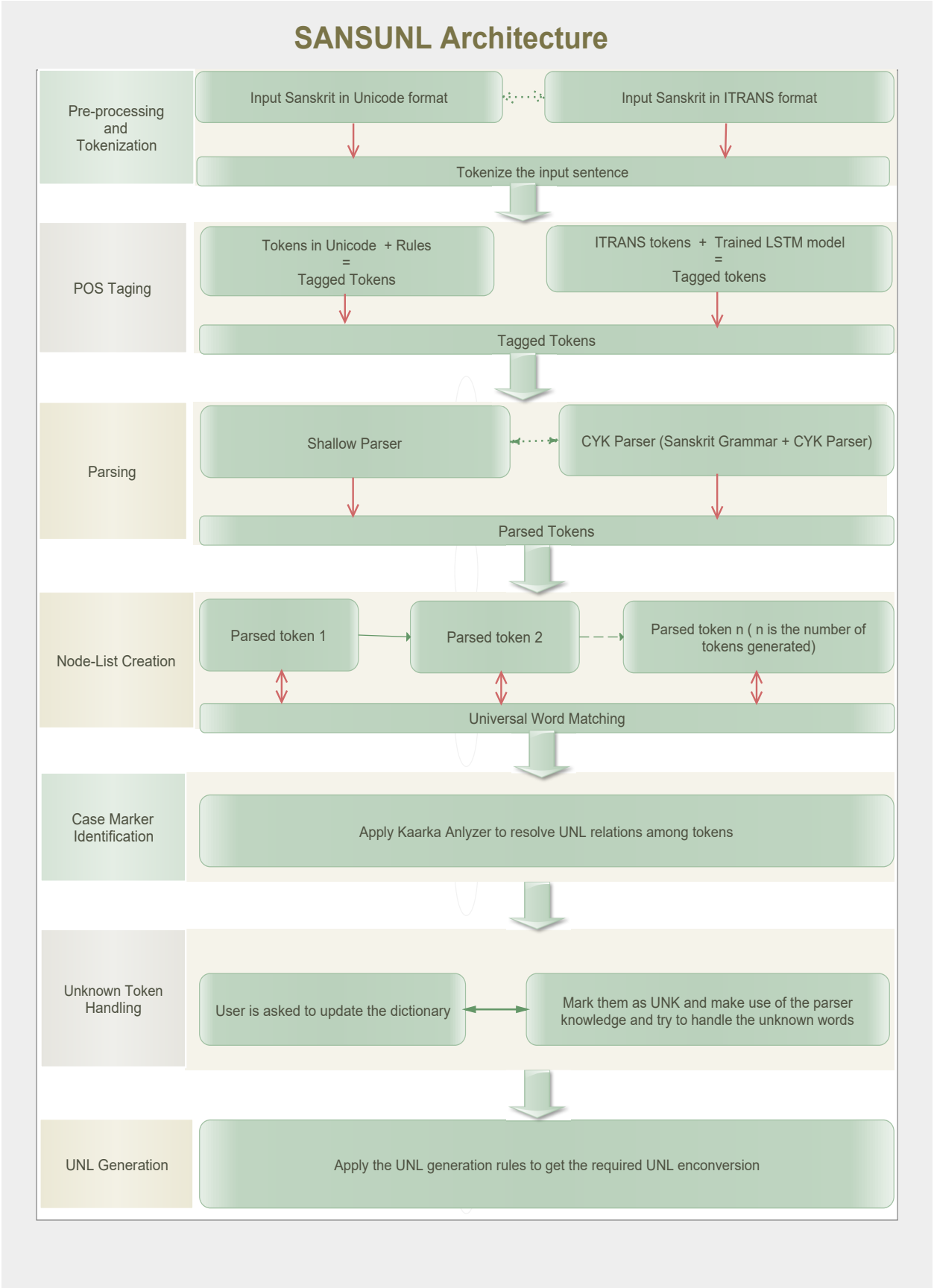

Figure 3.1: Architecture of SANSUNL

3.2 Pre-processing Layer

The first layer of the proposed architecture is pre-processing layer. This layer converts Sanskrit text taken as unicode to Indian Language Transliteration (ITRANS) or ITRANS to unicode and then tokenize the sentence. The input to the system can be given either in unicode format or ITRANS format. The ITRANS text is converted into unicode and the unicode text is converted into ITRANS using online "Sanscript tool" [149] . The input text is tokenized by using the regular expression and the StringTokenizer class of java with space as delimiter. The individual tokens are stored into 1-dimensional array.

3.3 POS Tagging Layer

The part-of-speech (POS) tagging plays an important role in developing an efficient MTS. It is the process of assigning different grammar roles like noun, verb, pronoun, proper-noun etc.to different words present in the sentence. It becomes more challenging in case of Sanskrit language due to less availability of Sanskrit text in digital form. The second layer of the proposed system performs the task of POS tagging. Several POS tagsets have been proposed [150, 70, 151, 152]. A comparison of such POS tagsets is presented in Table 3.1 which is based on well defined criteria that includes application to Sanskrit specific or common to many languages, fine grained or coarse grained analysis, flat or hierarchical structure, multi-lingual support and their basis for tagging.

Table 3.1: POS Tagset Comparison

	IIT / ILMT	JPOS	LDC-IL	IL-POSTS	CPOS
Common / Sanskrit	Common	Sanskrit	Common	Common	Sanskrit
Fine / Coarse Grained	Coarse	Fine	Fine	Fine	Coarse
Flat / Hierarchy	Flat	Flat	Flat	Hierarchy	Flat
Base	Penn Tree Bank	Paninian Grammar	ILMT	EAGLES	ILMT + JPOS
Multi-lingual Support	Yes	No	Yes	Yes	No
Number of Tages	26	134	26	7 (Cat) + 11 (Attributes)	28

POS tagsets discussed in Table 3.1 are available at:

https://www.sketchengine.eu/tagset-indian-languages/,

http://sanskrit.jnu.ac.in/corpora/JNU-Sanskrit-Tagset.htm,

http://www.ldcil.org/standardsTextPOS.aspx,

http://sanskrit.jnu.ac.in/corpora/MSRI-JNU-Sanskrit-Tagset.htm

and http://sanskrit.jnu.ac.in/cpost/post.jsp.

IL-POSTS Sanskrit tagset [153] has been selected for the proposed translation system after analyzing various POS tagsets in Table 3.1. Proposed system has adopted two strategies for POS tagging: stemmer based and neural network based tagging.

3.3.1 Stemmer based Tagging

Stemming is a process of removing morphological and inflectional endings from the words to get base form of a word. In first strategy, a stemmer has been proposed to stem the Sanskrit words and then corresponding rule from the rule-base has been applied to find out category of a word in the sentence. Proposed stemmer consists of 774 suffices and 23 prefixes which are further classified into three categories given below:

- The first category is of proper noun with 120 suffices.
- The second is of noun other than proper noun with 552 suffices.
- The third category consists of verb having 102 suffices.

To find the valid word and then to obtain the word category with case, number, person and gender information, a set of tagged Sanskrit words and Sanskrit to English dictionary have also been used in the stemming process. Further, output of the proposed POS tagger is compared with the existing Sanskrit analyser [154]. If more than one tags are obtained then the selection of correct tag is done based on tags of precedent word and successor word tag.

3.3.2 Neural Network based Tagging

The application of neural network on POS tagging is still a challenging task. In the proposed system, two Long Short Term Memory (LSTM) models are used on the tagged Sanskrit data-

set. The tagged data-set consists of approximately 0.4 million word entries. Fields in this data-set consists of Sanskrit words in ITRANS format with grammatical categories (noun, verb and pronoun) along with types and attributes. Sanskrit words with their grammatical category and attributes are extracted from this data-set using Python's XML parser and stored in the form of python record files. The proposed POS tagger is having four modules and are represented in Figure 3.2.

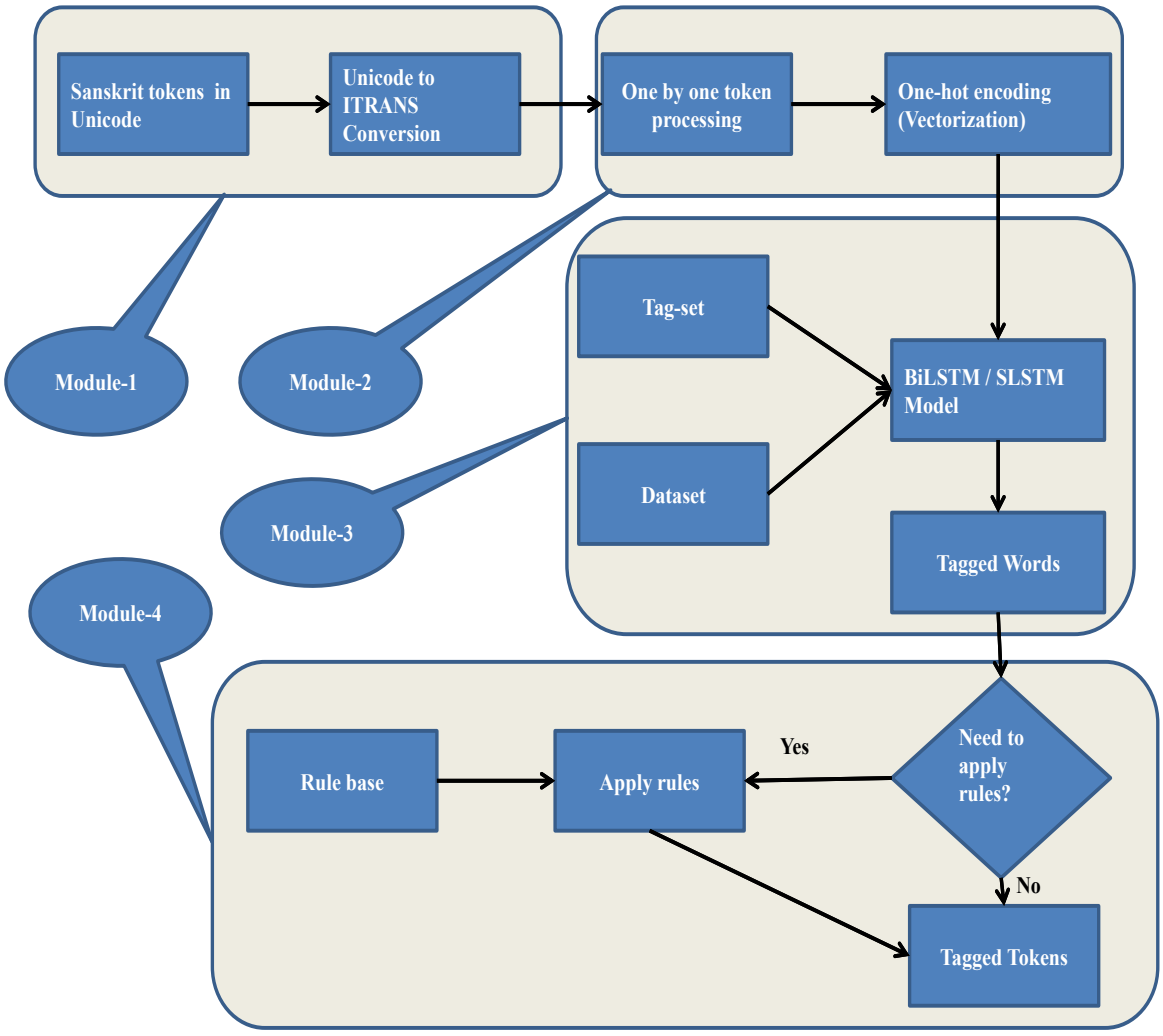

Figure 3.2: POS Tagger using LSTM

(i) **Module-1**

In this module, Sanskrit tokens in Unicode format are first converted into ITRANS format. The reason behind this is the availability of dataset in ITRANS. So to proceed

further the Sanskrit tokens need to be converted into ITRANS format if not already in that format.

(ii) **Module-2**

In this module, the tokens are accepted one by one and converted into vector form using one-hot encoding scheme [155].

(iii) **Module-3**

In this module, the dataset is divided into 80-20 ratio for training and testing purpose. Ten models have been built each for BiLSTM as well as SLSTM configuration. The switching among ten models is done as follows:

Model 1 – It predicts the word as noun, pronoun or verb.

Model 2 – If word is predicted as noun then this model predicts the gender.

Model 3 – If word is predicted as noun then this model predicts the case.

Model 4 - If word is predicted as noun then this model predicts the number.

Model 5 - If word is predicted as pronoun then this model predicts the gender.

Model 6 - If word is predicted as pronoun then this model predicts the case.

Model 7 - If word is predicted as pronoun then this model predicts the number.

Model 8 - If word is predicted as verb then this model predicts the verb root.

Model 9 - If word is predicted as verb then this model predicts the number.

Model 10- If word is predicted as verb then this model predicts the person.

After performing the training and testing of the models, the encoded tokens are forwarded to the models.

(iv) **Module-4**

If the output of the Module-3 is still ambiguous then the rules are applied to resolve any such ambiguity and the final tagged tokens as output are obtained.

3.4 Parsing

The tagged words obtained from previous layer as output are passed as input to the next layer i.e. parsing layer. In order to obtain the syntactic information about the sentence like subject, predicate, object etc. two approaches have been used for the parsing .These approaches are :

- Shallow Parsing
- CYK Parsing

These parsing technique are explained in the next sub-sections.

3.4.1 Shallow Parsing

In this parsing approach, a set of Sanskrit rules and word endings are used to perform the parsing. Sanskrit sandhi rules are applied in reverse to remove the word endings and then Sanskrit case markers rules are applied to find different roles (subject, object, verb and their person, number, gender information) of words in the sentence.

3.4.2 CYK Parsing

In this parsing strategy, a context-free grammar is designed for the Sanskrit language processing. Existing CYK parsing algorithm [156] is used to generate the parse tree for the Sanskrit grammar.

Sanskrit Grammar

$$G = \{N, \quad \sum, \quad P, \quad S\} \tag{3.1}$$

where

N={S, NP(obj), Predicate, NP(conj) } //set of Non-terminal symbols ,

$\sum = \{NP(subj), \quad VP, \quad Conj, \quad NP(Ind\_obj)\}$ //set of Terminal symbols ,

P is the set of production rules.

P= {

$$
\begin{array}{lll}
S & \rightarrow NP(subj)\,Predicate & | \; NP(conj)\,Predicate \\
NP(obj) & \rightarrow (obj)NP(Ind\_obj) & | \; NP(Ind\_obj)\;NP(obj) \\
Predicate & \rightarrow NP(obj)\,VP & \\
NP(conj) & \rightarrow NP(subj)\,Conj & | \; NP(subj)NP(conj)
\end{array}
$$

} S=S // start symbol.

Since the CYK parser uses only Chomsky Normal Form (CNF) of the CFG grammar. So the CFG grammar is converted into CNF form as follows:

$$G_1 = \{N_1, \quad \sum_1, \quad P_1, \quad S\}. \tag{3.2}$$

Where

$N_1 = \{S, \quad NP(obj), \quad Predicate, \quad NP(conj), \quad V, \quad X, \quad A, \quad B\}$ is the set of Non-terminal symbols

$\sum_1 = \{NP(subj), \quad VP, \quad Conj, \quad NP(Ind\_obj)\}$ //set of Terminal symbols

P_1 is the set of production rules.

P_1 ={

$$
\begin{array}{lll}
S & \rightarrow X\;\; Predicate & | \; NP(conj)\,Predicate \\
NP(obj) & \rightarrow NP(obj)A & | \; A\;\; NP(obj) \\
Predicate & \rightarrow NP(obj)\,V & \\
NP(conj) & \rightarrow X\;\; B & | \; X\;\; NP(conj) \\
X & \rightarrow NP(subj) & \\
A & \rightarrow NP(Ind\_obj) & \\
V & \rightarrow VP & \\
B & \rightarrow Conj &
\end{array}
$$

}

CYK Parsing Table

Proposed Sanskrit grammar is implemented using CYK Parser [157]. CYK parsing is done in a triangular form for any input string of length ‘m’ and grammar with ‘p’ non-terminals. The worst case time complexity of the CYK parser is $O(m^3)$ and space complexity is $O(m^2)$

[158], which is better than other parsing algorithms in worst case scenario. The process of CYK parsing is discussed as follows:

a) Initially tagged words are given as input.

b) Create a matrix of size [N, N] where N is the number of tokens in the sentence.

c) Fill the diagonal cells of the matrix with the mapped grammar's variables and terminals in the same order as tokens are present in the sentence.

d) If right side of the production rule can be partitioned into two parts then write the variable present at left side in that production rule of grammar at position [i,j]. The first part is present at [i,x] where x>i and x<j and second part is present at [y,j] where y<j and y>i.

e) Whenever there is more than one possibility, CYK implementation considers the one which is discovered at the later stage (the last one overwrites all previous reduction decisions in case of any overlapping).

f) At last, convert the CYK matrix into an actual tree by beginning from start symbol of grammar present at [0, N] and tracing children at each point.

If the input sentence is processed successfully by the proposed grammar, then Sanskrit parse tree is generated with the help of proposed Algorithm 1 from the parsing table and if not then control goes to section 3.4.1.

3.5 Node-List Creation and Universal Word Matching Layer

In this layer, a node list is created from the parsed text that has been generated in the previous layer. Each node consists of Sanskrit tagged word with corresponding English equivalent word. The node list also consists of syntactic and / or semantic attributes that are obtained from previous layer and will be updated in next layer. Figure 3.3 shows the structure of node

Algorithm 1: ParseTree Generation from Parsing Table

Input: Matrix M of order n*n , where n is the number of words in the input sentence
Output: Node list with Left, Parent and Right nodes

for $(i \leftarrow 0$ ***to*** $n-1)$ **do**
    Write Principle diagonal elements of the matrix M as leaf nodes.
    $Leaf[i] \leftarrow M(i,i)$
    $i \leftarrow i+1$
Take root variable to indicate the root of the tree.
Take three 1-D arrays L, P and R of size n-1 for storing Left, Parent and Right child of the tree.
Take a temporary variable temp and initialize it with value true.
Initialize m to 0;
$m \leftarrow 0$
$temp \leftarrow true$
for $(i \leftarrow 0$ ***to*** $n-2)$ **do**
    $L[m] \leftarrow Leaf[i]$
    for $(j \leftarrow i+1$ ***to*** $n-1)$ **do**
        if $(M(i,j) \neq NULL \wedge temp = true)$ **then**
            $//CellM(i,j) is not empty$
            make $M(i,j)$ as parent node of $L[m]$
            $P[m] \leftarrow M(i,j)$
            if *(P[m]='S')* **then**
                root=P[m]
            if $(j=i+1)$ **then**
                Make $Leaf[j]$ as the right node of the tree
                $R[m] \leftarrow Leaf[j]$
                $m \leftarrow m+1$
            else
                Make $M(i+1,j)$ as the right node of the tree
                $R[m] \leftarrow M(i+1,j)$
                $m \leftarrow m+1$
            $temp \leftarrow false$
        else
            if $(M(i,j) \neq NULL) \wedge (temp = false)$ **then**
                $//Cell$ M(i,j) is not empty
                $L[m] \leftarrow P[m-1]$
                $P[m] \leftarrow M(i,j)$
                if *(P[m]='S')* **then**
                    root=P[m]
                $R[m] \leftarrow Leaf[j]$
                $m \leftarrow m+1$
        $j \leftarrow j+1$
    $i \leftarrow i+1$
    $temp \leftarrow true$
for $(j \leftarrow 0$ ***to*** $n-2)$ **do**
    return (L,P,R)

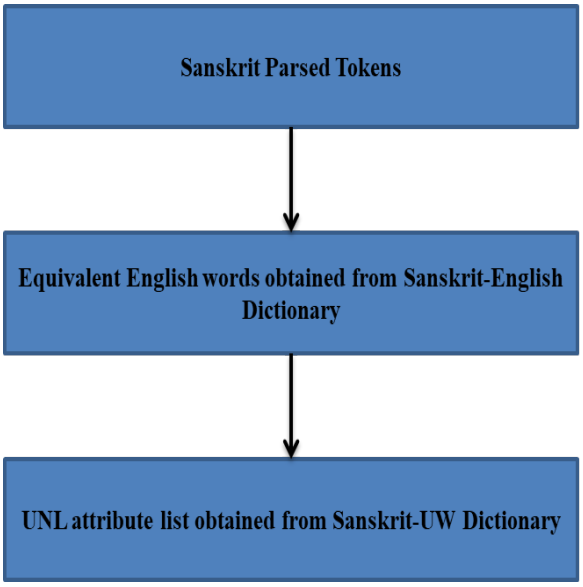

Figure 3.3: Structure of Node List

list. Each node is selected one by one from the node list and are searched in the Sanskrit-UW dictionary. There may be multiple entries for one word in the dictionary for depicting different aspects of a word in the sentence. To resolve the ambiguity among multiple entries and for selecting the correct word, grammatical attributes obtained from previous layer (POS tagger and parser) are used. Then updation of attributes in node list is performed after selecting correct word from dictionary. This process is repeated until all nodes in list are processed completely. If the node word does not exist in the dictionary then the user is asked to update the dictionary by marking the node word as UNK (unmatched word).

3.6 Case Marker Identification

The Sanskrit language is a morphological rich language. Unlike English, the preposition in Sanskrit are identified with the help of Kaarkaa (case) and are associated as suffix with the word. In this layer kaarkaa analyser have been used for identification of different role of words in the sentence and this information is used to resolve the UNL relations among nodes. The resolved UNL relations have been stored into relation table with corresponding links to the nodes.

3.7 Unmatched Word Handling Layer

The words for which no corresponding entry has been found in the UW dictionary are termed as unmatched words and are marked as “UNK”. In this layer to handle such words either a user is asked to update the dictionary or the grammatical attributes obtained from parser will be used to resolve UNL relations for such words.

3.8 UNL Expression Generation Layer

This layer is used to generate UNL expressions for the input Sanskrit sentence. After successfully resolving all the UNL relations among different UW’s, a set of approximately 1500 rules is applied to generate the UNL expressions for the input Sanskrit text. Once the initial node list is ready, two analysis windows are considered as First Analysis Window (FAW) and Second Analysis Window (SAW) for generating UNL expression. Following steps are used for generating the UNL expressions:

1. The first node is considered as part of the FAW and the next node as part of the SAW.

2. The processing will be done from FAW to SAW node by node and for each node required translation rule is searched in the rule base.

3. A rule is fired from the rule base depending upon the grammatical attributes of the node obtained from the language parser and is used for UNL relation resolution and extracting UNL attributes. The rule base updates the node list and the windows are also updated. If no rule is fired in the rule base, then go to step 5.

4. With this updated node list and windows, go to step 2. If the list consists of only one node then stop processing further by adding .@entry attribute along with corresponding number, person, tense /gender attributes depending upon the type of node whether it is a verb node or noun node.

5. If there exists no rule for the current nodes in the list, then perform simple right shift operation one node at a time and go to step 2 with this updated windows.

Chapter 4

Implementation of the Proposed SANSUNL System

This chapter provides implementation of the proposed system. Chapter starts with the working of proposed system followed by enconversion rule base, Sanskrit-UW dictionary and datasets used for the proposed system. Section 4.2 presents implementation of the proposed system by taking example of simple as well as complex sentences. Section 4.3 and section 4.4 presents the language divergence and translation among Sanskrit to English language respectively.

4.1 Working of proposed Sanskrit to UNL Enconverter System: SANSUNL

The flow chart shown in Figure 4.1 describes the step by step working of the proposed Sanskrit EnConverter system. The Sanskrit text is taken as input in the unicode / ITRANS format. The words in the input sentence are separated using word splitter module.The separated tokens are then sent to POS tagger.

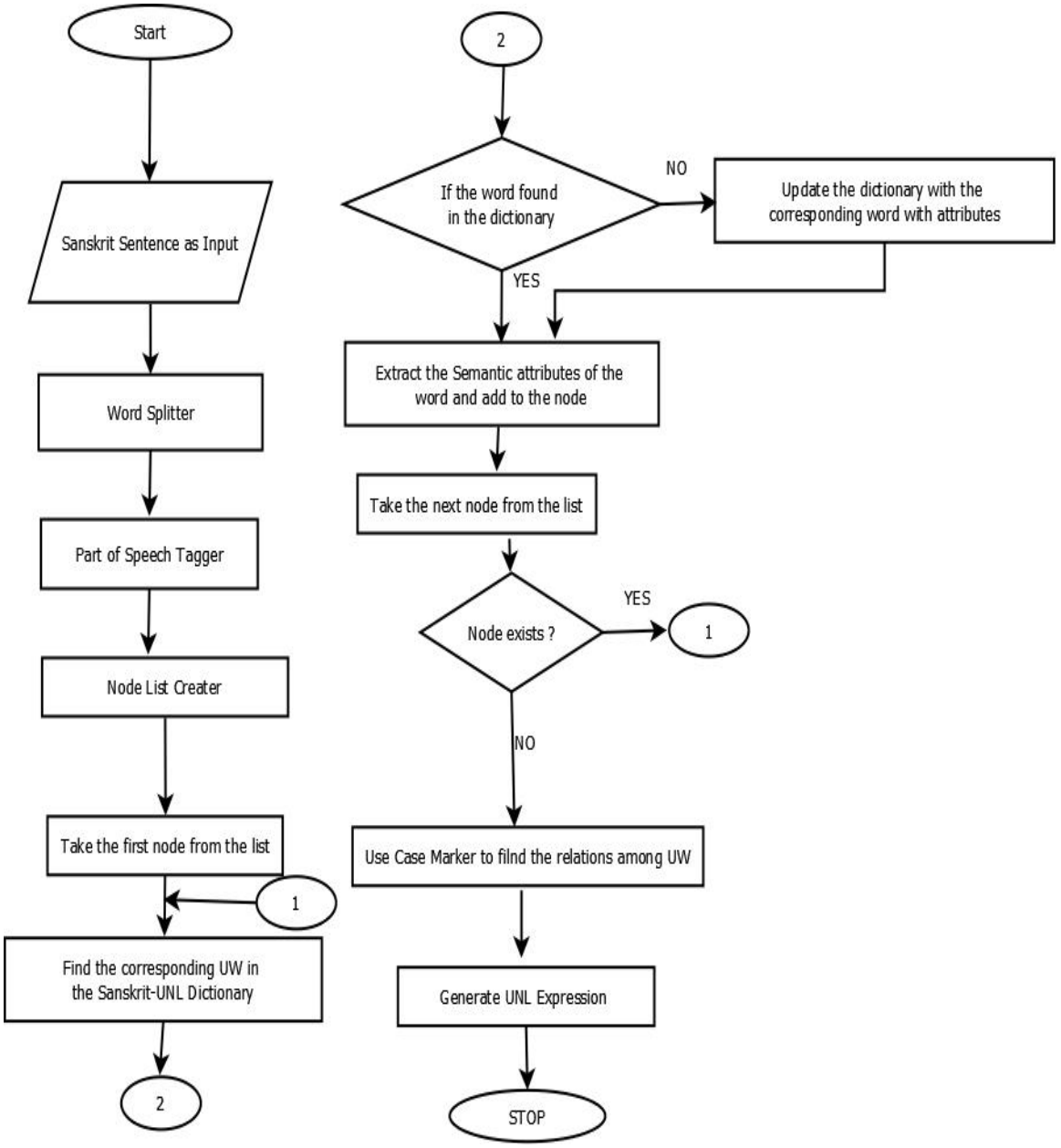

Figure 4.1: Working of Sanskrit EnConverter system

After performing the task of POS tagging and parsing, the node list is prepared. The nodes from the node list are matched in Sanskrit-UW dictionary. The semantic and syntactic attributes are attached to each nodes from the dictionary. The proposed system is using sliding window mechanism [46] [47] for processing the input text. The node list is processed from left to right. It uses two sliding windows as first analysis (AW) and condition windows (CD) as shown in Figure 4.2 for further processing. The AW window is surrounded by CD window.

The Enconversion rules (translation rules) are then applied based on the condition available in CD window to resolve the UNL relations among the nodes and after that sliding window is shifted towards right side. If no rule is fired then in that scenario, case marker module will be used to resolve the UNL relations and then windows will be updated. The application of enconversion rules may lead to addition or removal of attributes to the node list. All the node are processed in the same manner. At the end of processing, only the root node remains in the node list. The syntactic/ semantic tree is generated for all nodes using UNL relations among them. From the tree corresponding UNL expressions are generated as final output.

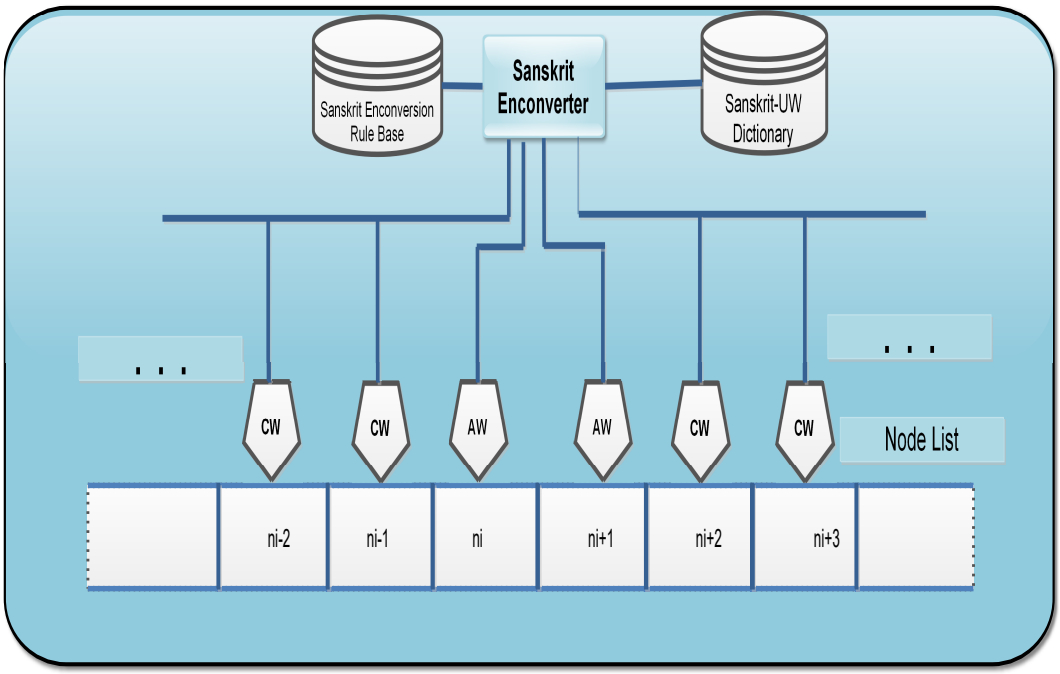

Figure 4.2: Basic Architecture of AW and CW for Sanskrit Enconverter

4.1.1 Enconversion Rule Base

The EnConversion rules are prepared as per the recommendations mentioned in UNL Specifications 2005. The format of the EnConversion rules for Sanskrit EnConverter is as follows:

<T>(<PRE>) ... <LN><RN><MID>(<SF1>)(<SF2>)....<PRI>

Where <T> denotes the type of rule

<PRE> <SF><MID> indicates the left, right, middle side of the Analysis Window

<PRI> indicates the priority of the Analysis/ EnConversion Rule which ranges from 0 to 255.

<LN> indicates the Left Node in the Analysis Window

<RN> Indicates the Right Node under the Analysis Window.

<LN> ::=< CONDITION1> :< ACTION1> :< RELATION1>

<RN> ::=< CONDITION2> :< ACTION2> :< RELATION2>

<CONDITION1> and <CONDITION2> are the conditions which needs to be satisfied for a rule to be fired i.e. these are the attributes (ATTR) whose presence or absence are responsible for any analysis rule to be fired. <ACTION1> and <ACTION2> are the actions taken in the form of addition or deletion of attributes from the nodes under Analysis Window.

<RELATION1> and <RELATION2> are used to resolve the UNL relations among the nodes under Analysis Window. The EnConversion Rules are divided into three main categories:

- Shift Left (L) or Shift Right (R)
- Left Modification (<) or Right Modification (>)
- Left Combination (+) or Right Combination (-)

A brief description of above types is given below:

- **Shift Left (L) or Shift Right (R)**

  This type of the rule is denoted by either ‘L’ or ‘R’. As the name indicates if no EnConversion rule is applicable on the nodes under Analysis Windows (AWs), then we simply shift the window on left or right side.

- **Left Modification (<) or Right Modification (>) rule**

  Here the type of the rule is denoted by either ‘<’ or ‘>’. As the name indicates these (< or >) rules are applicable when either right node modifies the left node under the analysis window or the left node modifies the right node under the analysis window. The modifier node (which performs the modification task) gets deleted from the node

list and the other node which is to be modified acts as the head of the remaining nodes in the node list.

- **Left Combination (+) or Right Combination (-)**

  This type of rule is denoted by either '+' or '-' . In this rule, either the right node is combined together with the left node known as left combination (+) or the left node is combined with the right node known as right combination (-) to form a composite node. In case of left combination rule the attributes of the left node are attached with the composite node for further processing with its position under Left Analysis Window (LAW). In second case of right combination, the attributes of the right node are attached with the composite node for further processing with its position under Right Analysis Window (RAW).

Examples of Enconversion rule is given below :

>(N,ANIMT,ACCTV,PLC,TIME,FRMT: null:obj) (V:+OBJRES:null)

>(PROPN,ANIMT,NOMINTV:null,agt)(V:+AGTRES:null)

>(N,INANI,NOMINTV:null:agt)(V:+AGTRES:null)

>(PRON,INANI,NOMINTV:null:modN,INANI:+MODRES:null

>(N,INANI,PLC,ABLTV,TIME,DUR,FRTRES:null,plf)(V:+PLFRES:null)

>(N,INANI,PLC,INSTV,TIME,DUR,FRTRES:null,src)(V:+SRCRES:null)

>(N,INANI,PLC,LOCTV:null:gol)(V:+GOLRES:null)

>(PRON,PERSPRON,ANIMT,NOMINTV:null:agt)(V:+AGTRES:null)

+{(N,INANI,CONJNCT,PLC,TIME,DUR:null,and)}{(N,INANI,PLC,TIME:null,null)}

+{(PROPN,ANIMT,NOMINTV:null,and)}{(PROPN,ANIMT,NOMINTV:null,null)}

>{(N,INANI,TIME,PLC:null:man)}{(V:+MANRES:null)}

>{(PRON,ANIMT:null:agt)}{(V,not,ability:+ .@not .@ability,AGTRES:null)}

>{(N,INANI,PLC,FRM:null:frm)}{(N:+FRMRES:null)}

>{(PROPN,INANI,PLC,NOMINTV:null:frm)}{(N,ANIMT,NOMINTV:+FRMRES:null)}

>{(N,ANIMT,POS,GENTV:null:pof)}{(N,POF:+POFRES:null)}

>{(PRON,POS:null,pos)}{(N,POF:+POSRES:null)}

>{(N,NOMINTV,RSN:null:rsn)}{(V:+RSNRES:null)}

>{(NUM,TIME,DUR:null:qua)}{(N,PLC:+QUARES:null)}

>{(N,CAG:null:ptn)}{(V:+PTNRES:null)}

>{(N,INANI,DATV:null:pur)}{(V:+PURRES:null)}

>{(PROPN,NOMINTV,NAM:null:nam)}{(N:+NAMRES:null)}

+{PROPN, ANIMT, NAM: +SAH :null}{[सह]:null:null}

>{N,INANI: +SAH :null} {[सह]:null:null}

>{N, INANI, SAH, PLC, : null: obj}{V:+OBJRS: null}

+{V:+ .@present .@sg .@3p:null}{[पठति]:null:null}

+{V:+ .@present .@du .@3p:null}{[पठतः]:null,null}

+{V:+@present .@pl .@3p:null}{[पठन्ति]:null:null}

+{V:+@present .@sg .@2p:null}{[पठसि]:null:null}

+{V:+@present .@du .@2p:null}{[पठथः]:null:null}

+{V:+@present .@pl .@2p:null}{[पठथ]:null:null}

+{V:+@present .@sg .@1p:null}{[पठामि]:null:null}

+{V:+@present .@du .@1p:null}{[पठावः]:null:null}

+{V:+@present .@pl .@1p:null}{[पठामः]:null:null}

+{V:+@past .@sg .@3p:null}{[अपठत्]:null:null}

+{V:+@past .@du .@3p:null}{[अपठताम्]:null:null}

+{V:+@past .@pl .@3p:null}{[अपठन्]:null:null}

+{V:+@past .@sg .@2p:null}{[अपठः]:null:null}

+{V:+@past .@du .@2p:null}{[अपठतम्]:null:null}

+{V:+@past .@pl .@2p:null}{[अपठत]:null:null}

+{V:+@past .@sg .@1p:null}{[अपठम्]:null:null}

+{V:+@past .@du .@1p:null}{[अपठाव]:null:null}

+{V:+@past .@pl .@1p:null}{[अपठाम]:null:null}

+{V:+@future .@sg .@3p:null}{[पठिष्यति]:null:null}

+{V:+ @future .@du .@3p:null}{[पठिष्यतः]:null:null}

+{V:+ @future .@pl .@3p:null}{[पठिष्यन्ति]:null:null}

+{V:+ @future .@sg .@2p:null}{[पठिष्यसि]:null:null}

+{V:+ @future .@du .@2p:null}{[पठिष्यथः]:null:null}

+{V:+ @future .@pl .@2p:null}{[पठिष्यथ]:null:null}

+{V:+ @future .@sg .@1p:null}{[पठिष्यामि]:null:null}

+{V:+ @future .@du .@1p:null}{[पठिष्यावः]:null:null}

+{V:+ @future .@pl .@1p:null} {[पठिष्यामः]:null:null}

+{V:+ .@present .@sg .@3p:null}{[गच्छति]:null:null}

+{V:+ .@present .@du .@3p:null}{[गच्छतः]:null,null}

+{V:+@present .@pl .@3p:null}{[गच्छन्ति]:null:null}

+{V:+@present .@sg .@2p:null}{[गच्छसि]:null:null}

+{V:+@present .@du .@2p:null}{[गच्छथः]:null:null}

+{V:+@present .@pl .@2p:null}{[गच्छथ]:null:null}

+{V:+@present .@sg .@1p:null}{[गच्छामि]:null:null}

+{V:+@present .@du .@1p:null}{[गच्छावः]:null:null}

+{V:+@present .@pl .@1p:null}{[गच्छामः]:null:null}

+{V:+@past .@sg .@3p:null}{[अगच्छत्]:null:null}

+{V:+@past .@du .@3p:null}{[अगच्छताम्]:null:null}

+{V:+@past .@pl .@3p:null}{[अगच्छन्]:null:null}

+{V:+@past .@sg .@2p:null}{[अगच्छः]:null:null}

+{V:+@past .@du .@2p:null}{[अगच्छतम्]:null:null}

+{V:+@past .@pl .@2p:null}{[अगच्छत]:null:null}

+{V:+@past .@sg .@1p:null}{[अगच्छम्]:null:null}

+{V:+@past .@du .@1p:null}{[अगच्छाव]:null:null}

+{V:+@past .@pl .@1p:null}{[अगच्छाम]:null:null}

+{V:+@future .@sg .@3p:null}{[गमिष्यति]:null:null}

+{V:+ @future .@du .@3p:null}[गमिष्यतः]:null:null}

+{V:+ @future .@pl .@3p:null}{[गमिष्यन्ति]:null:null}

+{V:+ @future .@sg .@2p:null}{[गमिष्यसि]:null:null}

+{V:+ @future .@du .@2p:null}{[गमिष्यथः]:null:null}

+{V:+ @future .@pl .@2p:null}{[गमिष्यथ]:null:null}

+{V:+ @future .@sg .@1p:null}{[गमिष्यामि]:null:null}

+{V:+ @future .@du .@1p:null}{[गमिष्यावः]:null:null}

+{V:+ @future .@pl .@1p:null}{[गमिष्यामः]:null:null}

<{V,+Present:+Past:null}{स्म:null:null)}

<{V,Past:+Present,+D not:null}{मा स्म:null:null}

<{N,INANI,ACCTV:++BASRES:null}{इव:null:bas}

<{N:+AND:null}{च:null:and}

<{PROPN:+AND:null}{च:null:and}

<{PRON:+AND:null}{च:null:and}

>{N:null:null}{N,+AND:null:ANDRES}

4.1.2 Sanskrit-Universal Word (UW) Dictionary

It is one of the key component of the UNL system. The Enconverter tool uses Natural language - Universal Word (L-UW) dictionary for the analysis of input natural language text. The main components of the dictionary are Head Word (HW), UW, Syntactic and Semantic attributes. The entries in the UNL dictionary are as per the UNL instruction format.

```
[HW]{ID}"UW" (ATTR1, ATTR2, \ldots..)<FLG, FRE, PRI>
```

where

HW: is the Head Word of the Sanskrit Language,

ID: is the Identifier, it may be left blank,

UW: is the Universal Word corresponding to the Sanskrit Head Word,

ATTR1,ATTR2: are the UNL Attributes of the UW,

FLG: is the language flag, for Sanskrit it is S,

FRE: is the frequency of the word to be used for the Enconversion and

PRI: priority of HW. PRI is the priority to be used to generate the Natural Language Code from UNL

The Sanskrit-UW dictionary consists of Sanskrit root word as HW with their corresponding UW in English followed by syntactic and semantic attributes. Sample of the Sanskrit - UNL dictionary is as follows:

[मोहन] Mohan (iof>person)"(PROPN,3S,M,NOMINTV,ANIMT,prsn,FAUNA)<S,0,0>;

HW = मोहन

UW = Mohan (iof>person)

ID = Blank {}

ATTR1 = PROPN grammar attribute is proper noun

ATTR2 = 3S indicates 3rd person singular

ATTR3 = M indicates the gender masculine

ATTR3 = NOMINTV is the case marker

ATTR4 = ANIMT indicates the Living being

ATTR5 = prsn indicates the person

ATTR6 = FAUNA is the habitat nature

FLG = S for Sanskrit

The syntactic attributes are: PROPN, M, 3S, NOMINTV and

The semantic attributes are = ANIMT, FAUNA

The Sanskrit–UNL dictionary consists of two main components as: Sanskrit specific (dependent) and Sanskrit free (independent). The first component includes Mohan (iof>person), ANIMT, FAUNA and the second component includes PROPN, 3S,

NOMINTV of the above example.

Several grammatical attributes are used in the Sanskrit-UNL dictionary. Some of the notations used to indicate the grammatical attributes are taken from Hindi-UNL dictionary. Some of the attributes are shown in table below:

Table 4.1: Grammatical Attributes

Grammatical Attribute	Explanation
PROPN	Proper noun
PRON	Pronoun
N	Noun
1S	1st person singular
2S	2nd Person Singular
3S	3rd Person Singular
1D	1st Person Dual
2D	2nd Person Dual
3D	3rd Person Dual
1P	1st Person Plural
2P	2nd Person Plural
3P	3rd Person Plural
M	Masculine gender
F	Feminine Gender
Ne	Neuter Gender
NOMINTV	Nominative Case
ACCTV	Accusative Case
INSTV	Instrumental Case
DATV	Dative Case
ABLTV	Ablative Case
GENTV	Genitive Case
LOCTV	Locative Case
VOCTV	Vocative Case
ANIMT	Animate
INANI	Inanimate
FAUNA	Fauna
FLORA	Flora
V	Verb
Present	Present Tense
Future	Future Tense
PAST	Past Tense
TRANSTV	Transitive Verb
INTRANSTV	Intransitive Verb
DITRANSTV	Di-transitive Verb

Table 4.1 Continued: Grammatical Attributes

PARASPD	Parasmenpadi Verb
ATMNPD	AAtmanepadi Verb
UBHPD	Ubhaypadi Verb
SPRTL	Spiritual
POS	Possessor
POF	Part of
VOA	Verb of action
VOO	Verb of occurrence
VOS	Verb of state
PLAC	Place
PHYSCL	Physical
ACTN	Action
TIM	Time
QAN	Quantity
QAL	Quality
MAN	Manner
NEGTV	Negative
NOTCH	No Change in Noun
DRNKBL	Drinkable
ACTN	Action
IMGRY	Imaginary
TTL	Title
INSCT	Insect
MACHN	Machine

Hindi-UW dictionary (IIT Bombay) has been used to create the Sanskrit-UW dictionary. The process of creation has been simple. While creating the Sanskrit-UW dictionary instead of Hindi head word, the equivalent Sanskrit head word is inserted in the Hindi-UW dictionary. The sample of the Sanskrit-UW dictionary is shown below:

[रामम्] {} "Ram(iof>person)"(PROPN,3S,M,ACCTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामेण] "Ram(iof>person)"(PROPN,3S,M,INSTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामाय] {} "Ram(iof>person)"(PROPN,3S,M,DATV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामात्] {} "Ram(iof>person)"(PROPN,3S,M,ABLTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामे] {} "Ram(iof>person)"(PROPN,3S,M, LOCTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामौ] {} "Ram(iof>person)"(PROPN,3D,M,NOMINTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामौ] {} "Ram(iof>person)"(PROPN,3D,M,ACCTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामाभ्याम्] {} "Ram(iof>person)"(PROPN,3D,M,INSTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामाभ्याम्] {} "Ram(iof>person)"(PROPN,3D,M,DATV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामाभ्याम्] {} "Ram(iof>person)"(PROPN,3D,M,ABLTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामयोः] {} "Ram(iof>person)"(PROPN,3D,M,LOCTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामाः] {} "Ram(iof>person)"(PROPN,3P,M,NOMINTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामान्] {} "Ram(iof>person)"(PROPN,3P,M,ACCTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामैः] {} "Ram(iof>person)"(PROPN,3D,M,INSTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामेभ्यः] {} "Ram(iof>person)"(PROPN,3P,M,DATV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामेभ्यः] {} "Ram(iof>person)"(PROPN,3P,M,ABLTV,ANIMT,prsn,FAUNA)<S,0,0>;

[रामेषु] {} "Ram(iof>person)"(PROPN,3P,M,LOCTV,ANIMT,prsn,FAUNA)<S,0,0>;

[वनम्] {} "Forest(icl>tree)"(N,3S,Ne,NOMINTV,ANIMT,tree,FLORA)<S,0,0>;

[वनम्] {} "Forest(icl>tree)"(N,3S,Ne,ACCTV,ANIMT,tree,FLORA)<S,0,0>;

[वनेन] {} "Forest(icl>tree)"(N,3S,Ne,INSTV,ANIMT,tree,FLORA)<S,0,0>;

[वनाय]" {} "Forest(icl>tree)"(N,3S,Ne,DATV,ANIMT,tree,FLORA)<S,0,0>;

[वनात्] {} "Forest(icl>tree)"(N,3S,Ne,ABLTV,ANIMT,tree,FLORA)<S,0,0>;

[वनस्य] {} "Forest(icl>tree)"(N,3S,Ne,GENTV,ANIMT,tree,POF,FLORA)<S,0,0>;

[वने] {} "Forest(icl>tree)"(N,3S,Ne,LOCTV,ANIMT,tree,PLC,FLORA)<S,0,0>;

[वने] {} "Forest(icl>tree)"(N,3D,Ne,NOMINTV,ANIMT,tree,FLORA)<S,0,0>;

[वने] {} "Forest(icl>tree)"(N,3D,Ne,ACCTV,ANIMT,tree,FLORA)<S,0,0>;

[मार्जारी] {} "cat((icl>thing) .@generic(N,3S,F,NOMNTV,ANIMT,FAUNA)<S,0,0> ;

[श्वेत] {} "white(aoj>thing)<S,0,0>;

[बिम्बैः] {} "spot"(N,3P,INSTV)<S,0,0>;

[कृष्णवर्णास्ति] {} "black(aoj>thing)"(V,3S,Present)<S,0,0>;

[अस्ति] {} "is(icl>existance)"(preposition)<S,0,0>;

[रविः] {} "Ravi(iof<person>)<PROPN,3S,M,NOMNTV,ANIMT,prsn,FAUNA)<S,0,0>;

[अम्रियत] {} "die(icl>occur)"(V,3S,PAST,PARASPD,TRANSTV,VOA)<S,0,0>;

[मम्] {} "I(icl>one person)"(PRON,PERSON,1S,GENTV,POS,ANIMT,my)<S,0,0>;

[पार्श्व] {} "have(aoj>thing,obj>thing)"(V,3S,Present,TRANSTV)<S,0,0>;

[अट्टालिका] {} "tarrace(icl>thing)"(N,3S,F,NOMNTV,INANI)<S,0,0>;

[भवनम्] {} "apartment(icl>thing)"(N,3S,Ne,NOMNTV,INANI)<S,0,0>;

[कृष्णाङ्गारम्] {} "ember(icl>thing)"(N,3S,Ne,NOMNTV,INANI)<S,0,0>;

[रक्तमासीत्] {} "red(aoj>thing)"(V,past,3S,paras)<S,0,0>;

[उष्णम्] {} "hot(aoj>thing)"(N,Ne,3S,nominative)<S,0,0>;

[नव] {} "nine(iof>large integer)"(N,M,INANI,ABS,NUM)<S,0,0>;

[होरायावत्] {} "hour(icl>period)"(N,Ne,3S,ABLTV,TIM)<S,0,0>;

4.1.3 Data-sets Used

For the implementation of the proposed system five data-sets DS1 to DS5 are used. The details of each data set is given below:

- Data-Set 1 (DS1)

  It is a tagged corpus of Sanskrit words. The size of the data set is approximately four lakh. This data set is extracted from XML-SL folder [85]. The SL folder consists of several xml files. For implementation of the proposed system, `SL_nouns.xml`, *SL_pronouns.xml* and *SL_roots.xml* files are used. The fields in the data-set consists of Sanskrit word in ITRANS format followed by the grammatical categories like noun, verb, pronoun with other attributes like number, person, gender, case and root form. The Sanskrit word with their grammatical category and attributes are extracted from the data-set using Python's XML parser and stored in the form of python record files. For training and testing of the neural network based POS tagger, DS1 is used along with a set of 774 suffixes and 23 prefixes. The sample of suffixes are shown in Figures 4.3, 4.4,

4.5, 4.6 and 4.7.

EDIT	ID	NOUN	NUM	GENDER	ENGEQ	SCAT	STATUS	HABITAT	HCAT
	1	अलका	S	स्त्रीलिंग	City of Kuber	स्वर्ग	Non-Living	-	-
	2	कैलासः	S	पुल्लिंग	Name of Mountain	स्वर्ग	Non-Living	-	-
	3	नलकूवरः	S	पुल्लिंग	Son of Kuber	स्वर्ग	Living	Land	Humans
	4	चैत्ररथम्	S	नपुंसकलिंग	The garden of Kubera	स्वर्ग	Non-Living	-	-
	5	कुबेरः	S	पुल्लिंग	Kuber	स्वर्ग	Non-Living	-	-
	6	अतिशयः	S	पुल्लिंग	Much or excessive	स्वर्ग	Non-Living	-	-
	7	सततम्	S	नपुंसकलिंग	Eternal or continually	स्वर्ग	Non-Living	-	-
	8	शीघ्रम्	S	नपुंसकलिंग	Swiftly	स्वर्ग	Non-Living	-	-
	9	रंहः	S	नपुंसकलिंग	Speed or velocity	स्वर्ग	Non-Living	-	-
	10	श्वसनः	S	पुल्लिंग	Air or wind	स्वर्ग	Non-Living	-	-

Figure 4.3: Sanskrit Noun Endings

EDIT	ID	SNO	RVERB	CAT	SCAT	CLASS	BASE	ENGEQ
	1	566	उक्ष्	प	T	1	उग	-
	2	567	उख्	प	T	1	उगा	-
	3	568	उङ्ख्	प	T	1	उङ्गा	-
	4	569	उच्	प	T	4	उजा	-
	5	570	उच्	प	T	1	उजा	-
	6	571	उच्	प	T	6	उजा	-
	7	572	उज्झ्	प	T	6	उ	-
	8	573	उञ्ज्	प	T	1	उञ्जा	-
	9	574	उञ्ज्	प	T	6	उञ्जा	-
	10	575	उठ्	आ	IT	1	उडा	-
	11	576	उठ्	प	T	1	उडा	-

Figure 4.4: Root form

EDIT	ID1	ID	ADVERB	ROOT
	1	1	सहसा	सहसा
	2	2	अकस्मात्	अकस्मात्
	3	3	आपातत:	आपातत
	4	4	अञ्जसा	अञ्जसा
	5	5	मुहु:	मुहु
	6	6	सकृत्	सकृत्
	7	7	अप्रयनत:	अप्रयनत
	8	8	प्रयनहीनत:	प्रयनहीनत
	9	9	प्रयनत:	प्रयनत
	10	10	समीपम्	समीपम्

Figure 4.5: Sanskrit Adjective Endings

EDIT	ID	SNO	MNAME	PADA	PERSON	NUM	MEND
	1	1	Imperative	P	1	S	आनि
	2	2	Imperative	P	1	P	आव
	3	3	Imperative	P	1	D	आम
	4	4	Imperative	P	2	S	तात्
	5	5	Imperative	P	2	P	तम्
	6	6	Imperative	P	2	D	त
	7	7	Imperative	P	3	S	तु
	8	8	Imperative	P	3	P	ताम्
	9	9	Imperative	P	3	D	अन्तु
	10	10	Imperative	A	1	S	ऐ

Figure 4.6: Sanskrit Mood Endings

EDIT	ID	SNO	TNAME	PADA	PERSON	NUM	TEND
	16	16	Present	A	3	S	ते
	17	17	Present	A	3	P	इते
	18	18	Present	A	3	D	अन्ते
	19	19	Imperfect	P	1	S	अम्
	20	20	Imperfect	P	1	P	व
	21	21	Imperfect	P	1	D	म
	22	22	Imperfect	P	2	S	स् - :
	23	23	Imperfect	P	2	P	तम्
	24	24	Imperfect	P	2	D	त
	25	25	Imperfect	P	3	S	त्

Figure 4.7: Sanskrit Tense Ending

- Data-Set2 (DS2)

  It is prepared from the data available at [29] from UC-A1. The data available under UC-A1 data-set consists of 50 English sentences taken from "Hare and tortoise story" with corresponding UNL expressions. To prepare the DS2, all 50 English sentences are first translated into Sanskrit manually and then the English sentences available in the data-set are replaced by these translated sentences. DS2 data-set is used to evaluate the proposed system.

- Data-Set3 (DS3)

  It is prepared from UC-A2 data-set available at [29]. UC-A2 data-set consists of 300 entries in English with corresponding UNL expressions. Out of 300 entries, first 70 are of cardinal numbers, next 50 are ordinal numbers, next 30 entries are of fractional or divisional part representations and rest 150 entries are simple sentence phrases. All the English entries are first translated manually into Sanskrit and then the corresponding UNL expressions are written for each entry.

- Data-Set4 (DS4)

  It is prepared from the Spanish language server data available at http://www.unl.fi.upm.es/english/fr_examples.htm. Again English sentences are translated into Sanskrit manually and then corresponding UNL expressions are written. 50 sentences are selected to cover maximun UNL relations. Table 4.2 shows DS4.

Table 4.2: Spanish Server Dataset (DS4)

S. No	Sanskrit	English	UNL Expressions
1	इदम् सङ्गणकम् अंग्रेजितः स्पेनिशभाषायाम् अनुवदति	This computer translates from English to Spanish	agt(translate(icl>do) .@entry, computer(icl>machine)) mod(computer(icl>machine), this) src(translate(icl>do) .@entry, english(icl>language)) gol(translate(icl>do) .@entry, spanish(icl>language))
2	रामः नियमान् खण्डयति	Ram breaks the rules	agt(break(icl>do) .@entry, "Ram")

Table 4.2 Continued: Spanish Server Dataset

			obj(break(icl>do) .@entry, rule .@generic .@pl)
3	विस्फोटेन वातायनानि अखण्डयत	The explosion broke the windows	agt(break(icl>do) .@entry .@past, explosion(icl>event) .@def) obj(break(icl>do) .@entry .@past, window .@def .@pl)
4	अहम् कारयानम् पूर्ववत्अकरवम् सुविधम् शीघ्रम् च	I repaired the car easily and quickly	agt(repair(icl>do) .@entry .@past, i(icl>person)) obj(repair(icl>do) .@entry .@past, car .@def) man(repair(icl>do) .@entry .@past, quickly) and(quickly, easily)
5	सः नविचारयितुम् समर्थ न च स्वप्नमंद्रष्टुम् शक्तः	He can't think nor dream	agt(dream(icl>do) .@entry .@not .@ability, he(icl>person)) and(dream(icl>do) .@entry .@not .@ability, think(icl>do) .@not .@ability)
6	रामः सीता च अयोध्यायाम्/ अयोध्ययोः वसतः	Ram and Sita live in Madrid	agt(live(icl>do) .@entry .@present, :01) and:01("Sita", "Ram")
7	मारजाराः मूषकेभ्यः अतिशेरते	Cats are nicer than rats	aoj(nice(aoj>thing) .@entry, cat(icl>thing) .@generic) man(nice(aoj>thing) .@entry, more) bas(more, rat(icl>thing) .@generic)
8	वायुयानम् नक्षत्रमिव दीव्यति	The airplane shines like a star	obj(shine(icl>occur) .@entry .@present, airplane(icl>thing) .@def) man(shine(icl>occur) .@entry .@present, like)

Table 4.2 Continued: Spanish Server Dataset

			bas(like, star(icl>thing) .@indef)
9	सः तव यन्त्रिका इव यन्त्रिका केष्यति	He will buy the same car as you	agt(buy(icl>do) .@entry .@future, he(icl>person)) obj(buy(icl>do) .@entry .@future, car) mod(car, same) bas(same, you(icl>person))
10	रामः सीतायै सुन्दरम् उपहरति	Ram gives a present to Sita	agt(give(icl>do) .@entry .@present, "Ram") ben(give(icl>do) .@entry .@present, "Sita")) obj(give(icl>do) .@entry .@present, present .@indef)
11	अहम् भवदर्थम् योत्स्यामि	I will fight for you	agt(fight(icl>do) .@entry .@future, i(icl>person)) ben(fight(icl>do) .@entry .@future, you(icl>person))
12	सीता रामेणसह चलति	Sita walks with Ram	agt(walk(icl>do) .@entry .@present, "Sita") cag(walk(icl>do) .@entry .@present, "Ram"))
13	मार्जारी श्वेत बिम्बैः कृष्णवर्णापि अस्ति	The cat is black with white spots	aoj(black(aoj>thing) .@entry .@present, cat .@def) cao(black(aoj>thing) .@entry .@present, spot .@pl) mod(spot .@pl, white(aoj>thing))
14	रविः कवितया सह अम्रियत	Ravi died with Kavita	cob(die(icl>occur) .@entry .@past, "Kavita") obj(die(icl>occur) .@entry .@past, "Ravi")
15	मम् पार्श्वे अट्टालिका भवनम् अस्ति	I have an apartment with terrace	aoj(have(aoj>thing, obj>thing) .@entry .@present, i(icl>person)) obj(have(aoj>thing,obj>thing) .@entry .@present, apartment(icl>thing))

Table 4.2 Continued: Spanish Server Dataset

			cob(have(aoj>thing,obj>thing) .@entry .@present, terrace(icl>thing))
16	कृष्णाङ्गारम् रक्तम् उष्णम् च असीत्	The ember was red as well as hot	aoj(red(aoj>thing) .@entry .@past, ember .@def) coo(red(aoj>thing) .@entry .@past, hot(aoj>thing))
17	सीता नव होरायावत् कार्यम् करोति	Sita works nine hours	agt(work(icl>do) .@entry, "Sita") dur(work(icl>do) .@entry, hour(icl>period)) qua(hour(icl>period), 9)
18	सः मम् अनुपस्थितौ आगतः	He came during my absence	agt(come(icl>do) .@entry .@past, he(icl>person)) dur(come(icl>do) .@entry .@past, absence(icl>state)) pos(absence(icl>state), i(icl>person))
19	रविः सोमवासरात् शुक्रवासरम् यावत् कार्यम् करोति	Ravi works from Monday to Friday	agt(work(icl>do) .@entry, "Ravi" .@def) man(work(icl>do) .@entry, monday(icl>time)) fmt(monday(icl>time), friday(icl>time))
20	अहम् जापानीयम् जनय अमिलम्	I met a man from Japan	agt(meet(icl>do) .@entry .@past, i(icl>person)) obj(meet(icl>do) .@entry .@past, man(icl>person) .@indef) frm(man(icl>person) .@indef, japan(icl>country))
21	सःअङ्कन्या समस्याम् समदधात्	He solved the problem using a pencil	agt(solve(icl>do) .@entry .@past, he(icl>person)) obj(solve(icl>do) .@entry .@past, problem .@def) ins(solve(icl>do) .@entry .@past, pencil(icl>thing) .@indef)

Table 4.2 Continued: Spanish Server Dataset

22	प्रकाशः रक्तवर्णे परिवर्तितः	The light changed to red	agt(change(gol>thing) .@entry .@past, light .@def) gol(change(gol>thing) .@entry .@past, red(aoj>thing))
23	अन्यान् लविवेण वियुक्ता	The piece was cut from edge	obj(cut(icl>do) .@entry .@past .@state, piece .@def) plf(cut(icl>do) .@entry .@past .@state, edge(icl>place) .@def)
24	सः प्रायशः रविम् स्मरति	He often thinks in John	agt(think(icl>do) .@entry, he(icl>person)) obj(think(icl>do) .@entry, "John") man(think(icl>do) .@entry, often)
25	सा अतीव सुन्दर	She is very beautiful	aoj(beautiful(aoj>thing) .@entry, she) man(beautiful(aoj>thing) .@entry, very)
26	अहम् त्वाम् कथाम् कथयिष्यामि	I will tell you the whole story	agt(tell(icl>do) .@entry .@future, i(icl>person)) ben(tell(icl>do) .@entry .@future, you(icl>person)) obj(tell(icl>do) .@entry .@future, story(icl>thing) .@def) mod(story(icl>thing), whole)
27	रविः मूलप्रतिमाम् कल्पयति	Ravi conceived a master plan	agt(conceive(icl>do) .@entry .@past, "Ravi") obj(conceive(icl>do) .@entry .@past, plan(icl>thing) .@in-def) mod(plan(icl>thing), master)
28	हिम खरडान् त्रुयम् आनय / मदर्थम् हिमखरडत्रयम् आनय	Bring me three blocks of ice	ben(bring(icl>do) .@entry .@imperative, i(icl>person))

Table 4.2 Continued: Spanish Server Dataset

			obj(bring(icl>do) .@entry .@imperative, block(icl>thing)) qua(block(icl>thing), 3)) mod(block(icl>thing), ice(icl>thing))
29	सीता टोक्योटावरम् अभ्रमत्	Sita visited the Tokyo tower	agt(visit(icl>do) .@entry .@past, "Sita") obj(visit(icl>do) .@entry .@past, tower(icl>thing) .@def) nam(tower(icl>thing), tokyo(icl>thing))
30	पोतः इब्रो नद्याम् निमज्जितः	The ship sunk in the Ebro river	obj(sink(icl>ocurr) .@entry .@past, ship .@def) plc(sink(icl>ocurr) .@entry .@past, river .@def) nam(river, "Ebro")
31	काष्ठफलकम् अन्यतः क्रियते	The table is moved	obj(move(icl>do) .@entry, table(icl>thing) .@def)
32	रुग्णः अरोग्यम् प्राप्तः	The patient is cured	obj(cure(icl>do) .@entry, patient(icl>person) .@def)
33	ग्रीष्मे हिमम् विलयति	Snow melts in summer	obj(melt(icl>occur) .@entry, snow(icl>thing) .@generic) tim(melt(icl>occur) .@entry, summer)
34	सः कलमेकम् धारयति सः कलमेकम् धारयति / तस्य पार्श्वे कलमम् अस्ति	He has a pen	obj(have(aoj>thing,obj>thing) .@entry, pen(icl>thing) .@indef) agt(have(aoj>thing,obj>thing) .@entry, he)
35	कार्गदम् अवघट्टे विदिर्णम्	The paper was cut in the middle	opl(cut(icl>do) .@entry .@past, middle(icl>place) .@def) obj(cut(icl>do) .@entry .@past, paper .@def)

Table 4.2 Continued: Spanish Server Dataset

36	किम् कारयानम् नीलवर्णा रक्ता व /यन्त्रिका नीला उत रक्ता	The car is red or blue	aoj(car .@def .@entry, red(icl>color)) or(red(icl>color), blue(icl>color))
37	अहम् प्रतिदिनम् द्वि आनयामि	I take two per day	agt(take(icl>do) .@entry, i(icl>person)) obj(take(icl>do) .@entry, 2) per(2 .@entry, day(icl>period))
38	अहम् महासंनसे पचामि	I cook in kitchen	plc(cook(icl>do) .@entry, kitchen(icl>thing) .@def) agt(cook(icl>do) .@entry, i(icl>person))
39	एतत् अधस्तात् रक्तम्	It is red on the bottom	aoj(red(aoj>thing) .@entry, it(icl>thing)) plc(red(aoj>thing) .@entry, bottom(icl>thing) .@def)
40	अहम् न्यूयार्कात् दूरवाणीम् करिष्यामि	I will call you from New York	agt(call(icl>do) .@entry .@future, i(icl>person)) ben(call(icl>do) .@entry .@future, you(icl>person)) plf(call(icl>do) .@entry .@future, new york(icl>place))
41	पक्षिणः पक्षाणि	Bird's wing	pof(wing(icl>body) .@entry, bird(icl>animal))
42	नरस्य सारमेयः	The man's dog	pos(dog(icl>thing) .@entry .@def, man .@def)
43	सा ग्रहात् आनयत्	She came from home	agt(come(icl>do) .@entry .@past, she(icl>person)) plf(come(icl>do) .@entry .@past, home(icl>place))
44	मम् पुस्तकम्	My book	pos(book(icl>thing) .@entry, i(icl>person))
45	भवान् इमम् द्रष्टुम् आगच्छेः	You should come to see this	agt(come(icl>do) .@entry .@should, see(icl>do))

Table 4.2 Continued: Spanish Server Dataset

			pur(come(icl>do) .@entry .@should, see(icl>do)) obj(see(icl>do), this
46	अहम् अनेक किलो मितम् कदली फलम् अक्रीणाम्	I bought several kilos of bananas	agt(buy(icl>do) .@entry .@past, i(icl>person)) obj(buy(icl>do) .@entry .@past, kilo(icl>unit) .@pl) qua(kilo(icl>unit) .@pl, several) mod(kilo(icl>unit) .@pl, banana .@generic)
47	सः स्वरोगवशः न अगच्छः	He did not go because of his illness	agt(go(icl>do) .@entry .@past .@not, he(icl>person)) rsn(go(icl>do) .@entry .@past .@not, illness(icl>thing)) pos(illness(icl>thing), he(icl>person))
48	सः कोषागरात् धनम् चोरयति	He steals money from banks	agt(steal(icl>do) .@entry, he(icl>person)) obj(steal(icl>do) .@entry, money) src(steal(icl>do) .@entry, bank(icl>thing) .@generic)
49	मध्याह्निके सूर्य प्रचण्डः जायते	The sun is full at noon	aoj(full(aoj>thing) .@entry, sun(icl>thing) .@def) tmf(full(aoj>thing) .@entry, noon(icl>time) .@def)
50	रामः न्यूयार्कात् शिकागो गच्छति	John goes to Chicago via New York	agt(go(icl>do) .@entry, "John") plt(go(icl>do) .@entry, chicago(icl>place)) via(go(icl>do) .@entry, new york(icl>place))

- Data-Set5 (DS5)

  It consists of 500 simple Sanskrit sentences. Figure 4.8 shows sample of DS5

data set.

1.बालकः पठति । a child studies.
2.अशोकः चलति ।ashok walks.
3.सुलेखा नमति ।sulekha bows down.
4.राम् श्यामः च तत्र् एवं क्रीडतः ।ram and shyam play there.
5.पिता पुत्रः च हसतः ।father and son laugh.
6.सिता गीता च तत्र गच्छतः ।seeta and geeta go there.
7.तौ अत्र् भ्रमतः ।they both roam here.
8.ते शीघ्रम् लिखन्ति ।they all write quickly.
9.अशोकः उद्याने क्रीडिष्यति ।ashok will play in the garden.
10.वयम् विद्यालयम् गमिष्यामः ।we all will go to the school.
11.आवाम् फलं खादिष्यामः ।we both will eat fruits.
12.रामस्य पत्नी सीता आसीत् ।sita was ram's wife.
13.लक्ष्मणः सीता च अपि वनं ।laxman and sita also went to the forest.
14.लक्ष्मणं शत्रुघ्नः च सुमित्र्याः पुत्रौ आस्ताम् ।laxman and shatrughan were sumitra's sons.
15.वने राक्षसाः अवसन् ।monsters lived in the forests.
16.बालकः पुस्तकम् पठति । child studys the book.
17.आवाम् जन्तुशालां गच्छावः । we both are going to the zoo.
18.तत्र आवाम् अनेकान् पशून् द्रक्ष्यावः । there we both will see various animals.
19.अत्र अल्पाहारगृहम् अपि भविष्यति ।here there will be canteen also.
20.अधुना विश्रामं करिष्यावः ।now we will take rest.
21.सीता क्रीडति ।sita plays.
22.अले वदति।aleksh says.
23.त्वम् उपविशसि |you sit.
24.ताः अष्टवादने क्रीडन्ति | They play at 8 o'clock.

Figure 4.8: DS5

In addition to the above data sets, the proposed system have also used Sanskrit to UW dictionary and Sanskrit to English dictionary for implementation.

4.2 Implementation

The implementation of the proposed system is demonstrated with examples. Two examples are used to show the implementation process by taking simple sentence and complex sentence.

4.2.1 Simple Sentence Implementation

This section is showing the implementation using a simple Sanskrit sentence containing only one subject, object and verb. The implementation is shown step by step as follows:

Example 1. SS: मोहनः विद्यालयम् गच्छति

ES: Mohan goes to school

1. Pre-Processing and Tokenization

   The ITRANS form of the above Devanagari text is as follows:

   ITRANS : mohanaH vidyAlayam gachChati

   The words from the sentences are tokenized using the regex class and String-Tokenizer class of java. The output after tokenization is :

   token1= मोहनः

   token2= विद्यालयम्

   token3=गच्छति

2. POS Tagging

   POS tagging of the above sentence is done using set of grammar rules and shallow parser as shown in Figure 4.9.

   मोहनः= मोहन पुं 1 एक

   Mohan= Mohan, m, nominative, sg.

विद्यालयम् = विद्यालय पुं 2 एक

To school=school m accusative sg

गच्छति=गच्छत् पुं 7 एककृदन्त गच्छत् नपुं 7 एककृदन्त

= गम् कर्तरि लट् प्र एक परस्मैपदीधातुः गम्लृँगणः भ्वादिः

Goes=go sv pres 3p sg transit v go class 1st

Where sg=singular, m=masculine, acc=accusative, pres=present Tense, n=neuter, transit=transitive, 1p=1st Person, 3p=3rd person, 2p=2nd person, nom=nominative case, sv=subject oriented verb, v= verb.

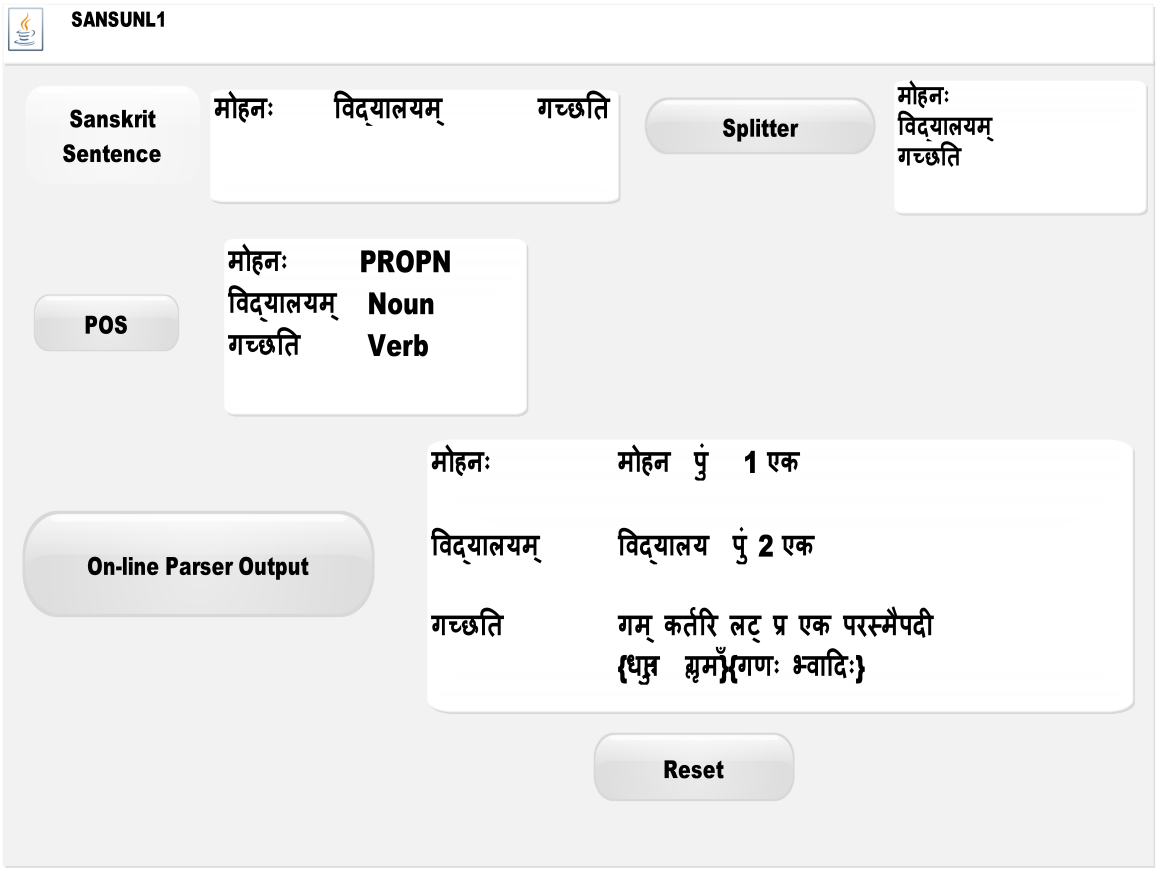

Figure 4.9: POS tagging and parsing

3. Node list Creation

After processing POS tagging and parsing, the node list is created as follows: node1 values = Sanskrit word =मोहनः, UW=null, POS=PROPN, syntactic /

semantic attributes= m, nominative, sg

node2 values = Sanskrit word=विद्यालयम् ,UW=null,POS=Noun, syntactic / semantic attributes=m, accusative, sg

node3 values= Sanskrit word= गच्छति, UW=null, POS=verb, syntactic / semantic=pres, 3p, sg, transit, 1st

4. Universal Word Matching

This module add UW and semantic attributes to the node list from Sanskrit-UW dictionary. Searching is done based on Sanskrit word with POS and syntactic/ semantic attributes obtained from POS tagging and parsing module. The output of this module is as follows:

node1 values:

Sanskrit word =मोहनः,

UW=Moha(iof>person),

POS=PROPN,

syntactic / semantic attributes= m, nominative, 3s, ANIMT, FAUNA

node2 values:

Sanskrit word=विद्यालयम्,

UW=school(icl>place),

POS=Noun,

syntactic / semantic attributes=m, accusative, 3S, INANI, PLC, phscl

node3 values:

Sanskrit word= गच्छति,

UW=go(icl>move),

POS=V,

syntactic / semantic=pres, 3p, sg, transit, bhavaadi, voa-act-bodly, 1st

5. Case Marker Identification and UNL Generation Module

This module is used to find the UNL relations among different nodes of the node list. To find the UNL relations Kaarka analyzer and the steps mentioned in section 3.8 are used.

The node-list is represented in between as "<<<" and ">>>" and nodes under observations are represented by square brackets as between "[" and "]". Initial position of nodes in the list is as follows:

<<<[मोहनः] [विद्यालयम्] गच्छति >>>

$$<<< [mohanaH][vidyAlayam]gachChati >>> \tag{4.1}$$

In the above case, no rule will be fired so shift right (R) operation will be applied to the sliding window to shift to right side. The new node list will be:

<<<मोहनः [विद्यालयम्] [गच्छति]>>>

$$<<< mohanaH[vidyAlayam][gachChati] >>> \tag{4.2}$$

Now the following rule will be fired :

$$> (Noun, ANIMT, ACCTV : null : obj)(V, : +OBJRES : null) \tag{4.3}$$

This rule will delete the left node from the list and retain right node in the list. The UNL relation "obj" is resolved between the nodes vidyAlayam and gachChati as follows:

$$obj(go(icl > move), school(icl > place)) \tag{4.4}$$

The new node list will become as:

<<<[मोहनः] [गच्छति]>>>

$$<<< [mohanaH][gachChati] >>> \tag{4.5}$$

$$> (PROPN, NOMINTV, ANIMT : null : agt)(V : +AGTRES : null) \tag{4.6}$$

Again the UNL relation “agt” will be resolved and the left node will be removed from the list as :

$$agt(go(icl > move), Mohan(iof > person)) \tag{4.7}$$

<<<[गच्छति]>>>

$$<<<[gachChati]>>> \tag{4.8}$$

Now only one node is left in the node list so this will be considered as the entry node and if it is a verb then add tense, person and number attributes along with “ .@” attribute to this node as follows:

$$go(icl > move).@Present.@3S.@entry. \tag{4.9}$$

The semantic graph for the above sentence is shown in Figure 4.10. The nodes are UW's and the arc labels are the relations among them.

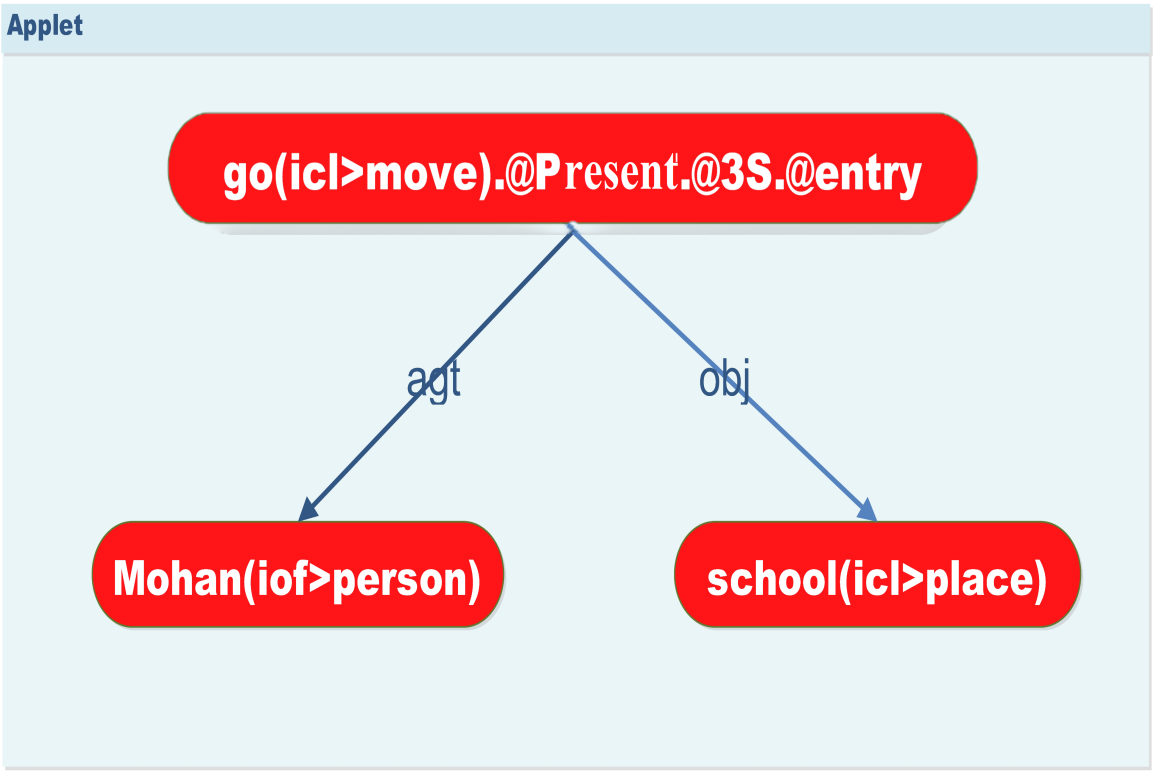

Figure 4.10: Semantic Graph

From semantic graph the UNL expressions can be easily extracted and the final output is shown in Figure 4.11.

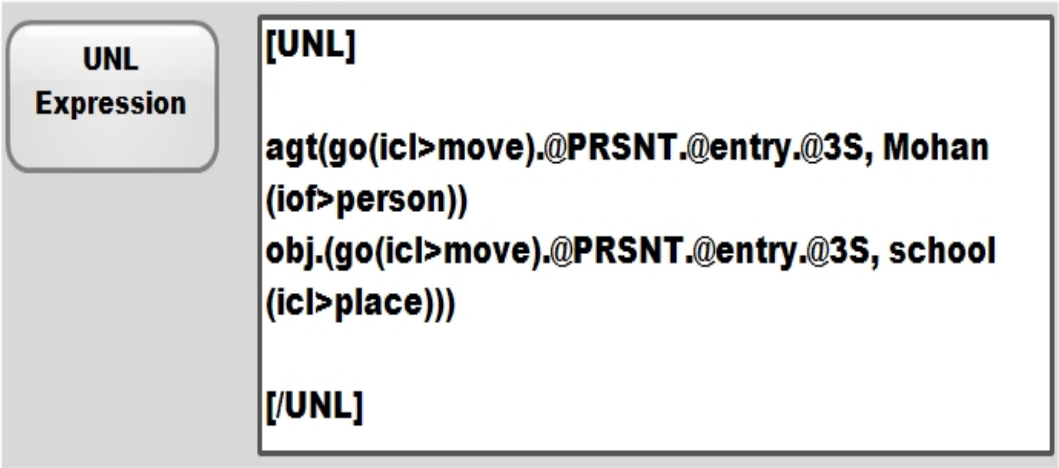

Figure 4.11: UNL Expression

4.2.2 Complex sentence implementation

This section is showing the step by step implementation of complex Sanskrit sentence having subject, verb, direct object, indirect object etc.

Example 2. Sanskrit SS: रामः ओदनम् चमसेन कपिलस्य थालिकायाः खादति

ITRANS : rAmaH odanam chamasena kapilasya thAlikAyAH khAdati

IAST: rāmaḥ odanam camasena kapilasya thālikāyāḥ khādati.

1. POS Tagging

   Figure 4.12 shows the POS tagging of Sanskrit tokens.

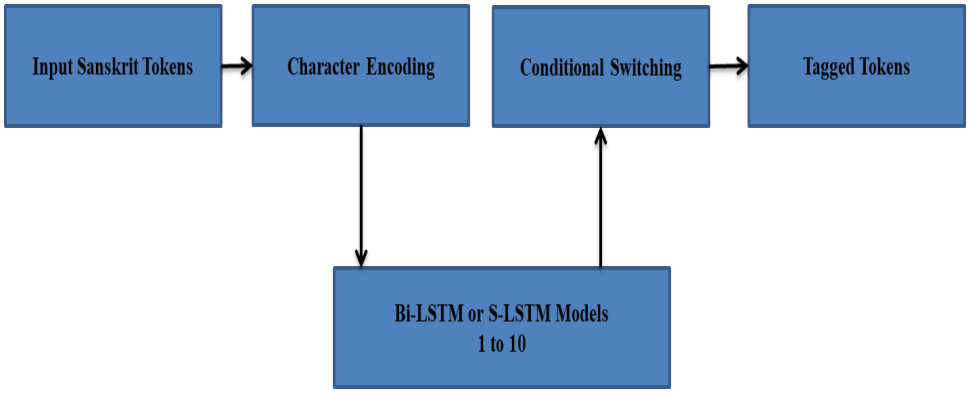

Figure 4.12: LSTM Based Tagging

One-hot embedding is used to do the character based embedding for the neural network [159]. Figure 4.13 depicts word categories as Noun, Pronoun and Verb with their attributes as gender, number, person and type used for tagging the tokens. Figure 4.14 shows tagged output for the sentence tokens.

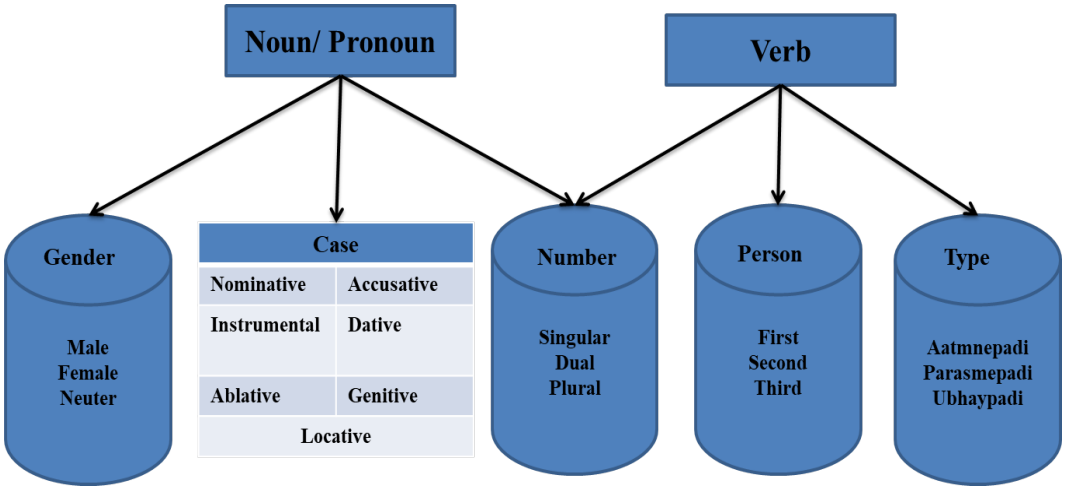

Figure 4.13: Word Categories and their Attributes

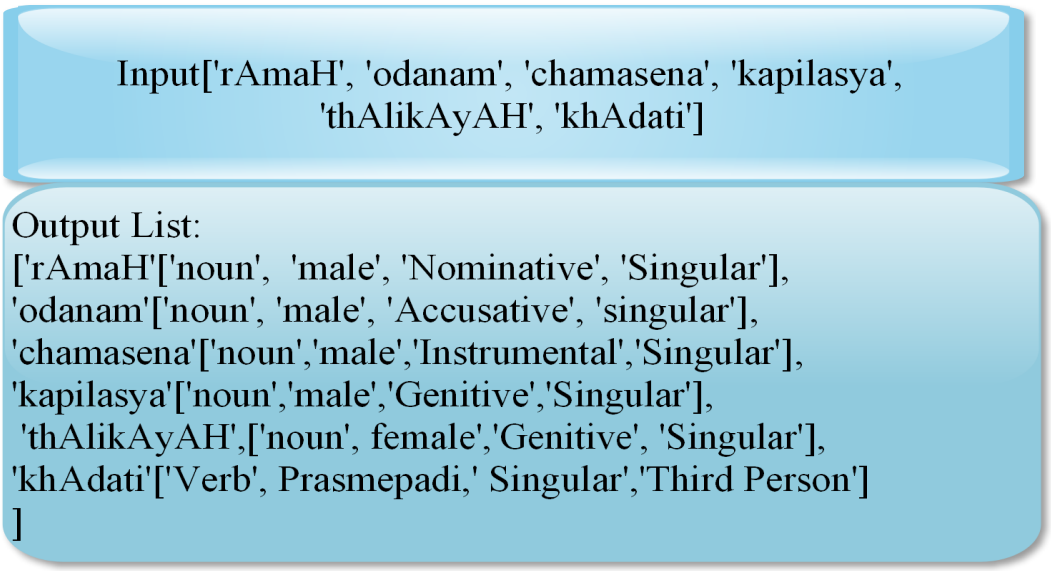

Figure 4.14: Tagged Tokens

2. Parsing

   The tagged tokens are converted back into Devanagari form and processed by the proposed Sanskrit grammar. Parse tree is generated to get the specific role of word in the sentence. Figure 4.15 shows the Sanskrit Parse Tree generated by the Sanskrit grammar.

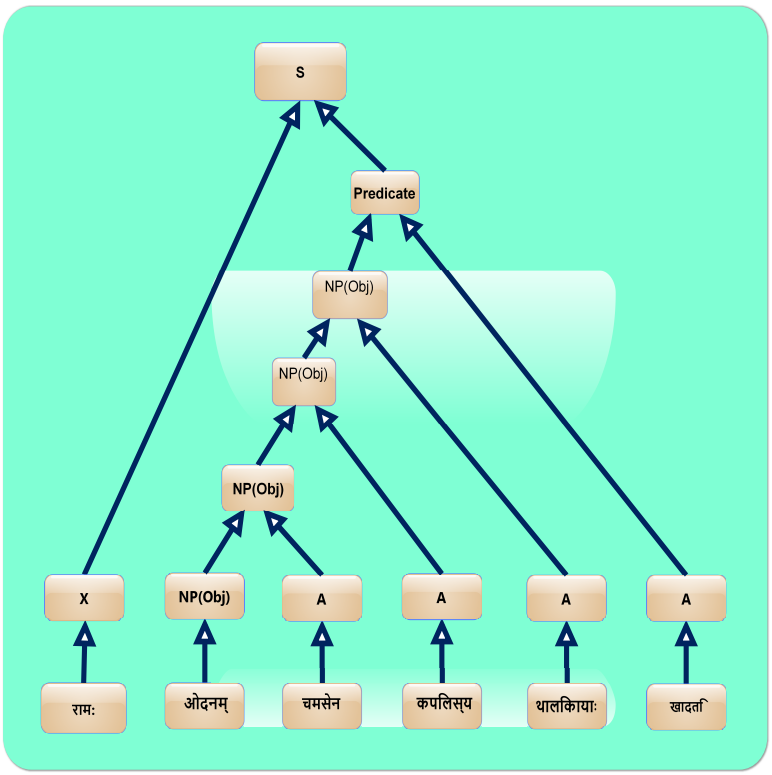

Figure 4.15: Sanskrit Parse Tree

3. Node list creation

   The node list created after the parsing phase is shown below:

   रामः - >Ram (noun, male, Nominative, Singular, X(NP(Sub))

   ओदनम् -> Rice (noun, male, Accusative, singular, Np(Obj))

   चमसेन ->Spoon (noun, male, Instrumental, Singular, Np(Ind_Obj))

   कपिलस्य -> Kapil's (noun, male, Genitive, Singular ,Np(Ind_Obj))

   थालिकायाः ->Plat (noun, female, Ablative, singular,vNp(Ind_Obj))

   खादति ->eat(verb, Prasmepadi, Singular, Third Person, Np(Ind_Obj))

   If any ambiguity persists then the Sanskrit grammar rules are used to disambiguate.

4. UW Dictionary Matching

   The attributes obtained after parsing phase have been used to identify the word

in the UW dictionary and the UNL attributes are added to the node list. Also the case marking is done with the help of Kaarka analyzer.

[रामः]"Ram" (N, M, Nomtv, 3S, ANIMT, FAUNA, X(NP(Sub))

[ओदनम्] "Rice" ((N, M, Acctv, 3S, ANIMT, FLORA, NP(Obj))

[चमसेन] "Spoon" ((N, M, Instrumental, 3S, INANI, Np(Ind_Obj))

[कपिलस्य "Kapil's" ((N, M, Genitive, 3S, ANIMT ,FAUNA, Np(Ind_Obj))

[थालिकायाः]"Plat" ((N, F, Abltv, 3S, INANI, KitUtensil, Np(Ind_Obj))

[खादति] "eat"(v,3S, Prasmepadi, Present, @entry, VOA, Np(Ind_Obj))

5. UNL expression generation

This is the final step in which the UNL expressions are generated. Nodes are scanned from left to right using a window size of two. The UNL relations are resolved using same process as was used in previous system, but with enhanced number of rules. The semantic graph of the above example is shown in Figure 4.16 below:

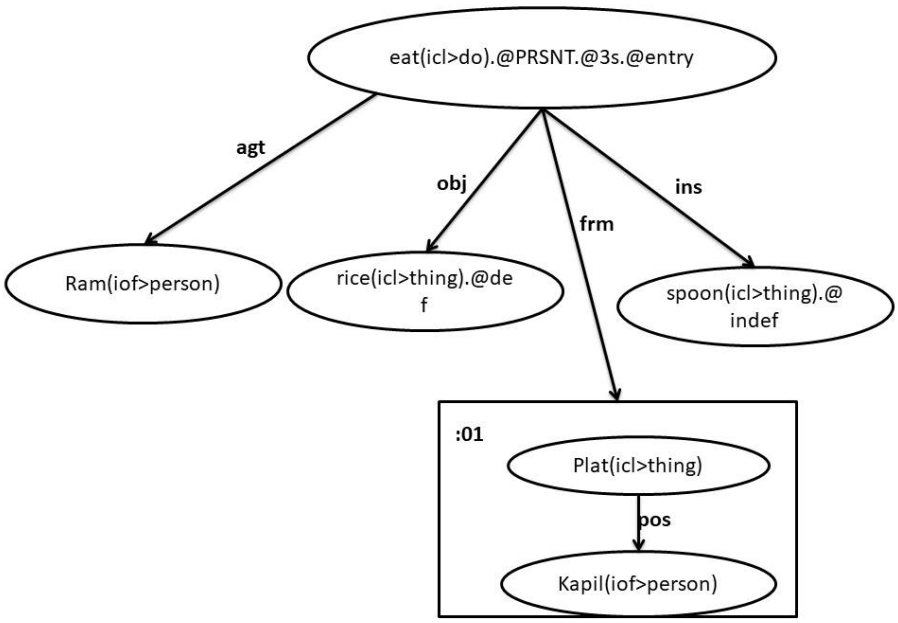

Figure 4.16: Semantic graph for Example2

UNL expressions generated from the semantic graph are as follows:

agt(eat(icl>do) .@entry .@present, Ram(iof>person))

obj(eat(icl>do) .@entry .@present, rice(icl>thing) .@def))

ins(eat(icl>do) .@entry .@present, spoon(icl>thing) .@indef))

frm(eat(icl>do) .@entry .@present, :01)

pos:01(Plat(icl>thing), Kapil(iof>person))

In this particular example only five phases have been used as there is no unmatched word found and case marking has been done during the universal word attribute extraction process.

4.3 Language Divergence among Sanskrit and English: Identification and Recommendation

It is necessary to understand divergence among the languages under consideration before starting the translation process. Dorr [160] has classified the language divergence problem into seven categories based on lexical and semantic attributes with their possible solutions. Figure 4.17 shows the divergence classification among Sanskrit and English languages.

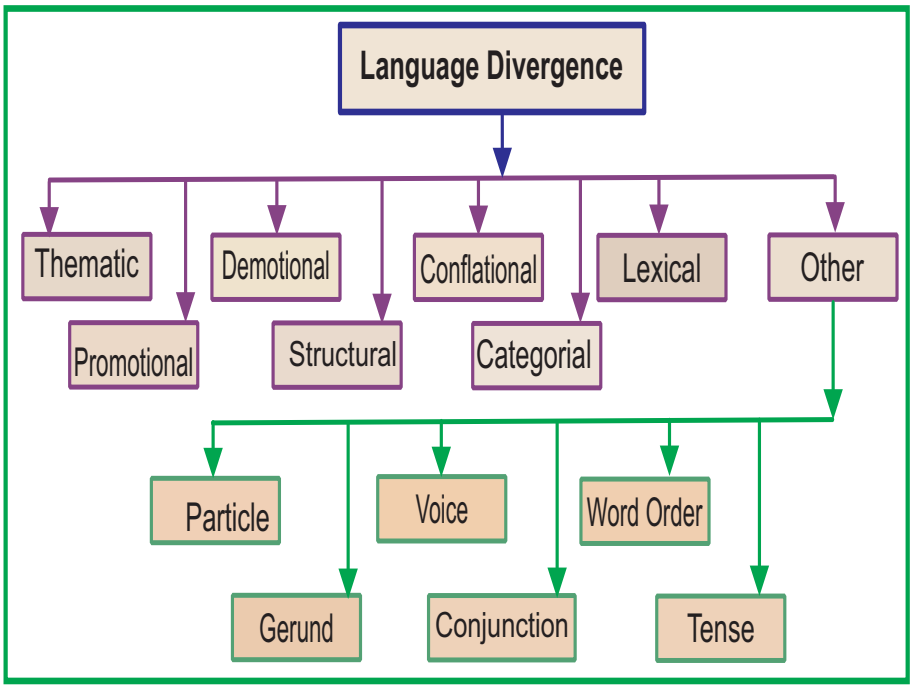

Figure 4.17: Language Divergence

Goya et al. [161] and Mishra et al. [162] also identified the language divergence among English and Sanskrit language which includes Dorr's classification and other divergence patterns also. [162] provided the recommended solutions for the divergence identified. Three types of information are required to provide a solution to any divergence: GLR, CLR and LCS. GLR and CLR are language independent whereas LCS stores the language-dependent information about lexical items. An exception in GLR or CSR or both in either of the language indicates the occurrence of lexical divergence.

Table 4.3 shows various types of divergence between Sanskrit and English language with possible recommendations during translation. In Table 4.3 "SS" denotes Sanskrit sentence and "ES" denotes the English sentence. The Sanskrit sentences are written in both unicode as well as in ITRANS format.

Table 4.3: Language Divergence among Sanskrit and English

No.	Divergence	Explanation	Example	Recommendations
1	Thematic divergence	It arises only in case of the logical subject. In Sanskrit to English, the thematic divergence occurs when the re-positioning of subject and object occurs concerning verb from Sanskrit to English translation.	(i) (a) SS: मह्यम् मधुर्म् रोचते(Mahyam Madhurm Rocate) (b) ES: Sweets are liked by me.	Re-positioning of argument with respect to given head.
2	Structural divergence	It occurs from Sanskrit to English when the realization of Noun in Sanskrit is done with vibhakti (preposition) whereas in English the vibhakti part is realized with or without a preposition.	(i) (a) SS: ऋहम् फलम् खादामि (Aham Phalam Khaadaami) (b) ES:I eat fruits (ii) (a) SS:ऋहम् विद्यालयम् गच्छामि(Aham Vidyaalayam Gacchaami) (b) ES: I go to school (iii) (a) SS: रामः सीताम् ददर्श (Raamah Siitaam Dadarsha) (b) ES: Ram saw Sita. In sentence i(a),ii(a) and iii(a) the noun फलम्, विद्यालयम् and सीताम् are realized with vibhakti in Sanskrit but in English i(b),ii(b) and iii(b) their equivalents are realized with (to school) or without (fruits, Sita) preposition .	Such divergence can be resolved by using * marker in the lexicon for such items to indicate that corresponding elements in English may need compositional realization.
3	Inflation Divergence	From Sanskrit to English the Inflational divergence occurs in which one Sanskrit word gets translated into more than one English words.Example (i) and (ii) shows such divergence.	(i) (a) SS:रामः पन्डितायते(Raamah Panditaayate) (b) ES: Ram behaves like a scholar. (ii) (a) SS: सीता पापच्चयते(Siitaa Paapaccayate) (b) ES: Sita cooks again and again. (Sita) (cooks again and again)	Proper attention is required for handling the compound words of Sanskrit in this case.
4	Categorical Divergence	Categorical divergence occurs from Sanskrit to English Translation as shown in example (i).	(i) (a) SS: रामः सीताया ईर्ष्यति (Raamah Siitaayaa Iirssyati) (b) ES: Ram is jealous of Site. As seen in (i)(b) Jealous is recognized as verbal in POS tagging in Sanskrit whereas in English it is adjective.	It is related to CSR not with the GLR. This issue can be resolved using CAT parameter in Sanskrit lexical entry not in English as suggested by [160].

Table 4.3 Continued: Language Divergence between Sanskrit and English language

5	Lexical Divergence	Lexical divergence arises due to other divergences as shown in examples (i) and (ii).	(i) (a) SS:सः विद्यालयम् गच्छति (Sah Vidyaalayam Gacchati) (b) ES: He goes to school. (ii) (a) SS:सः विद्यालयात् आगच्छति (Sah Vidyaalayaat Aagacchati) (b) ES: He comes from school. As shown in example (i) and (ii) sentences for Sanskrit गच्छ is translated by English verb "go", whereas in the next sentence Sanskrit verb आ + गच्छ is translated by another verb "come" which shows the lexical divergence among Sanskrit and English language.	Such divergence problem could be solved by proper selection of the lexical item.
6	Particle Divergence	In Sanskrit the participle is formed by adding the suffix 'tum'(तुम्) directly to any root verb and in English it is formed by adding 'to' before the verb. (i) In Sanskrit the dative case may be used in place of using 'तुमुन्' infinitive and replaces 'to' with 'for' in English sentence as shown in example (i),(ii),(iii) and (iv). (ii) Use of Particle 'sma'. When added to present tense it converts into Past tense as shown in example (v) and (vi). (iii) When ma sma मा स्म is added to past tense in Sanskrit it gets converted into present as shown in example (vi) and (vii).	(i) (a) SS:सीता वक्तुम् इच्छति (Siitaa Vaktum Icchati) (b) ES: Sita wants to speak. (ii) (a) SS:सीता वचनाय इच्छति (Siitaa Vacanaaya Icchati) (b) ES: Sita desires for speaking. (iii) (a) SS:नगरम् संरक्षितुम् रक्षकाः सन्ति (Nagaram Samrakssitum Rakssakaah Santi) (b) ES: The police are to protect the city. (iv) (a) SS: नगरसंरक्षणाय रक्षकाः सन्ति (Nagarasamrakssannaaya Rakssakaah Santi) (b) ES: The police is for the protection of the city. (v) (a) SS:अहम् खादामि (Aham Khaadaami) (b) ES: I am eating (vi) (a) SS:अहम् खादामि स्म (Aham Khaadaami Sma) (b) ES: I was eating (vii) (a) SS:त्वम् मूर्खम् अभवः (Tvam Muurkham Abhavah) (b) ES: You became a fool. (viii) (a) SS: त्वम् मूर्खम् मा स्म भवः (Tvam Muurkham Maa Sma Bhavah) (b) ES: Do not be a fool.	1. if SS contains 'tum'(तुम्) suffix then add 'to' before the corresponding verb in English for making infinitive particle or if SS contains dative case then add 'for' before the corresponding verb in English. 2. if (SS= verb + स्म sma) then use past tense in English else use the present tense. 3. if (SS=verb +मा स्म (maa sma)) then the verb gets converted into present tense.

Table 4.3 Continued: Language Divergence between Sanskrit and English language

7	Gerund Divergence	Gerund divergence occurs at the time of gerund realization in both Sanskrit and English Language. When the single subject is performing two tasks, then to show the completion of the first task before the commencement of the second one, we use 'क्त्वा orल्यप्' past participles instead of using 'and then' phrase as shown in example (i) and (ii).	(i)	(a) SS: बालकाः तेषां अभ्यासं कृत्वा विद्यालयम् गच्छन्ति(Baalakaah Tessaam Abhyaasam Krtvaa Vidyaalayam Gacchanti) (b) ES: Boys go to school having done their study.	To resolve this divergence, whenever we see 'त्वा' or ल्यप्' ending with a verb,then use 'having + 3rd form of the verb' in English.
			(ii)	(a) SS:आगराम् गत्वा वयं ताजमहलम् द्रक्ष्यामः (Aagaraam Gatvaa Vayam Taajamahalam Drakssyaamah) (b) ES: Having gone to Agra, we will see the Tajmahal.	
8	Voice divergence	In Sanskrit, there are three types of voices: Active (कर्तरि), Passive (कर्मणि) and Bhave (भावे) whereas in English we have only two voices: Active and Passive. Divergence is shown in example i,(ii) and (iii) sentences.	(i)	(a) SS: भक्ता: देवीम् पूज्यन्ति (Bhaktaa: Deviim Puujyanti)(Active) (b) The devotees worship the goddess.	To translate Bhave voice of Sanskrit in which subject and object both will be in an instrumental case, and the verb will always be singular+ 3rd person to generate English equivalent we should use 'being + 3rd form of the verb'.
			(ii)	(a) SS:भक्तैः देवी पूज्यते (Bhaktaih Devii Puujyate) (Passive) (b) The goddess is worshiped by the devotees.	
			(iii)	(a) SS: भक्तैः देव्या पूज्यते (Bhaktaih Devyaa Puujyate)(Bhave) (b) The goddess is being worshiped by the devotees.	
9	Conjunction Divergence	In Sanskrit the conjunction like ' vaa, athvaa' plays multiple roles in sentence formation. The sentences (i) to (iii) in example shows the divergence ('or' in English).	(i)	(a) SS:रवि पटियालाम् गतवान अस्ति वा चन्डीगढम्(Ravi Pattiyaalaam Gatavaana Asti Vaa Canddiigadddham) (b) ES: Ravi has gone either to Patiala or to Chandigarh.	To identify such divergence, we have to see in Sanskrit sentence the occurrence of 'vaa, athvaa' and the solution is by using "Either-or ‖ or" in English equivalent.
			(ii)	(a) SS:हिन्दीम् वा संस्कृतम् वदतु ‖ हिन्दीम् अथवा संस्कृतम् वदतु (Hindiim Vaa Samskrtam Vadatu ‖ Hindiim Athavaa Samskrtam Vadatu) (b) ES: Speak in Hindi or Sanskrit.	
			(iii)	(a) SS:किम सीता गच्छति वा आगच्छति(Kima Siitaa Gacchati Vaa Aagacchati) (b) ES: Does Sita going or coming?	
			In the above sentences 'vaa, athvaa' acts as coordinate conjunction ('either-or ‖ or' in English) in Sanskrit which joins two clauses as in sentence (i) and two phrases as in sentence (ii). Moreover, we have 'vaa and athvaa' in Sanskrit for single 'or' in the English language.		

Table 4.3 Continued: Language Divergence between Sanskrit and English language

10	Word order Divergence	Although the Sanskrit language is a free word order language, i.e., Subject (S), Object (O), Verb (V) could come at any position, but in case of interrogative sentences this free word order characteristic creates a problem as shown in Example 1 sentence.	(i)	(a) SS:किम् रामः पठति ? रामः किम् पठति ? रामः पठति किम् ? (Kim Raamah Patthati ? Raamah Kim Patthati ? Raamah Patthati Kim ?) (b) ES: Is Ram studying? What is Ram studying? Is Ram studying?	Reordering of the target sentence as per the English language format.
11	Tense Divergence	A single Sanskrit sentence is realized by two sentences in English which show divergence in both directions as shown in example (i)	(i)	(a) SS: (Sah Vidyaalayam Gacchati) (सः विद्यालयम् गच्छति) (b) ES: He goes to school. (c) ES: He is going to school.	Table 4.4 of 4.3.1 is used to recommend a solution for such divergence.

4.3.1 Target Language Generation Rule (TLGR) Base

This section provides the TLGR and covers three voice of Sanskrit language with corresponding English language equivalent. Table 4.4 shows tabular representation of three voice and ten tenses of Sanskrit with rules to generate English equivalent translation.

Table 4.4: Target Language Generation Rule Base

लट् लकार (Present Tense)			
Active Voice			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form
First ((उत्तम)	Singular	Agrees in Person and Number with Subject	V_1 (Present Indefinite) or am + V_1 + ing (Present Continuous)
	Dual/ Plural		are + V_1 +ing (Present Continuous)
Second (मध्यम)	Singular/ Dual/ Plural		V_1 (Present Indefinite) or are + V_1 +ing (Present Continuous)

Table 4.4 Continued: Target Language Generation Rule Base

Third (प्रथम)	Singular		s/es + V_1 (Present Indefinite) or is + V_1 +ing (Present Continuous)
	Dual/ Plural		V_1 (Present Indefinite) or are + V_1 +ing (Present Continuous)
Passive Voice			
Person (Object)	Number (Object)	Sanskrit Verb	English Verb Form
First ((उत्तम)/ Second (मध्यम)/ Third (प्रथम)	Singular	Agrees in Person and Number with Subject	Is+ V_3 (Passive form of Present Indefinite)
	Dual/ Plural		are+ V_3 (Passive form of Present Indefinite)
Abstract Voice			
Person (Object)	Number (Object)	Sanskrit Verb	English Verb Form
First ((उत्तम)/ Second (मध्यम)	Singular (Object)	Verb will be in singular and 3rd person form.	Is+ being+ V_3 (Passive form of Present Continuous)
Third (प्रथम)	Dual/ Plural (Object)		are+being+ V_3 (Passive form of Present Continuous)
(Past Tense) लङ् ,लिट् and लुङ्) लकार			
Active Voice			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form
First ((उत्तम)	Singular	Agrees in Person and Number with Subject	V_2 (Past Indefinite) or was + V_1 +ing (Past Continuous)

Table 4.4 Continued: Target Language Generation Rule Base

	Dual/ Plural		were + V_1 +ing (Past Continuous)
Second (मध्यम)	Singular/ Dual/ Plural		V_2 (Past Indefinite) or were + V_1 +ing (Past Continuous)
Third (प्रथम)	Singular		V_2 (Past Indefinite) or was + V_1 +ing (Past Continuous)
	Dual/ Plural		were + V_1 +ing (Past Continuous)
Passive Voice			
Person (Object)	Number (Object)	Sanskrit Verb	English Verb Form
First ((उत्तम) / Second (मध्यम)/ Third (प्रथम)	Singular	Agrees in Person and Number with Subject	was+ V_3 (Passive form of Past Indefinite)
	Dual/ Plural		were+ V_3(Passive form of Past Indefinite)
Abstract Voice (Subject and Object both will be in Instrumental Case)			
Person (Object)	Number (Object)	Sanskrit Verb	English Verb Form
First ((उत्तम)/ Second (मध्यम)/ Third (प्रथम)	Singular	Verb will be in singular and 3rd person form only.	was+ being+ V_3 (Passive form of Past Continuous)
	Dual/ Plural		were+being+ V_3 (Passive form of Past Continuous)
लिट् लकार (Past Perfect)			
Active Voice			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form

Table 4.4 Continued: Target Language Generation Rule Base

First ((उत्तम)/ Second (मध्यम)/ Third (प्रथम)	Singular / Dual/ Plural	Agrees in Person and Number with Subject	had+V_3 (Past Perfect)
लुङ् लकार (Aorist/ Past Perfect Continuous)			
Active Voice			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form
First ((उत्तम)/ Second (मध्यम)/ Third (प्रथम)	Singular / Dual/ Plural	Agrees in Person and Number with Subject	Had+been+V_3
लृट् लकार (Simple Future)			
Active Voice			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form
First ((उत्तम)	Singular / Dual/ Plural	Agrees in Person and Number with Subject	will/shall+ V_1
Second (मध्यम)/ Third (प्रथम)	Singular / Dual/ Plural		will + V_1
लुट् लकार (Future Continuous)			
Active Voice			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form
First ((उत्तम)	Singular / Dual/ Plural	Agrees in Person and Number with Subject	will/shall + be+ V_1+ing
Second (मध्यम)/ Third (प्रथम)	Singular / Dual/ Plural		will + be+ V_1+ing
लोट् लकार (Imperative Mood)			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form

Table 4.4 Continued: Target Language Generation Rule Base

First ((उत्तम)/ Second (मध्यम)/ Third (प्रथम)	Singular / Dual/ Plural	Agrees in Person and Number with Subject	Let+V_1 +! must+ V_1 +! (In case of Command) Or Please + V_1(In case of Request) Or Can+ V_1 + ? (in case of question)
विधिलिङ् लकार (Potential Mood)			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form
First ((उत्तम)/ Second (मध्यम)/ Third (प्रथम)	Singular / Dual/ Plural	Agrees in Person and Number with Subject	May+V_1(Possibility) Should+V_1(Advice) Should+be+V_3 (Appropriatness) Should+have+V_3 (possibility) Should+not+ V_1(Notice)
आशिलिङ् लकार(Benedictive Mood)			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form
First ((उत्तम)/ Second (मध्यम)/ Third (प्रथम)	Singular / Dual/ Plural	Agrees in Person and Number with Subject	May+V_1(blessing)
लृङ् लकार (Conditional Mood)			
Person (Subject)	Number (Subject)	Sanskrit Verb	English Verb Form

Table 4.4 Continued: Target Language Generation Rule Base

First ((उत्तम)/ Second (मध्यम)/ Third (प्रथम)	Singular / Dual/ Plural	Agrees in Person and Number with Subject	If+V_1

4.4 Sanskrit to English Translation

This section shows the architecture of proposed Sanskrit to English translator. Figure 4.18 shows six modules in architecture of the proposed system.

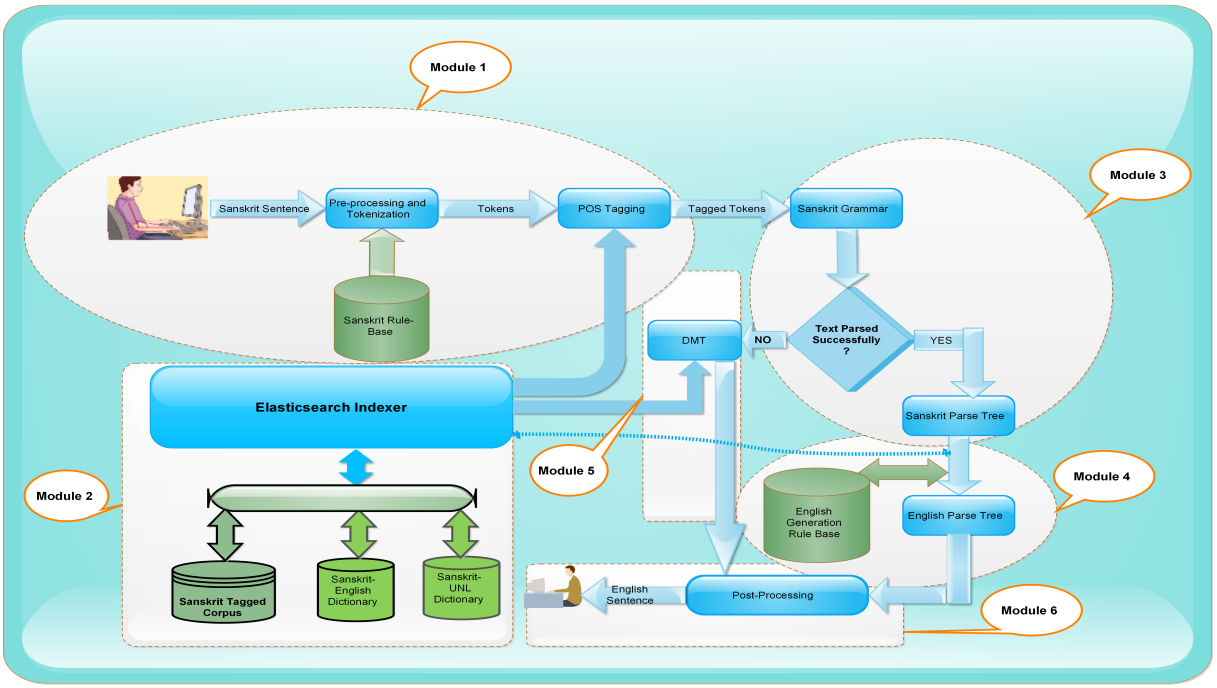

Figure 4.18: Architecture of the proposed Sanskrit to English MT System

1. Module-1

   This module performs the pre-processing and part-of-speech tagging of the source language sentence.

   (a) Source Language pre-processing

   Although Sanskrit is a free word order language, the proposed system uses the SOV order for processing Sanskrit sentences. In this phase, Sanskrit

sentence is taken as input using unicode encoding scheme. The input sentence is checked for the SOV grammar structure and restructured into the SOV format if it does not follow the required grammar structure. The reformatting is done using the Kaarka Analysis (Case structure)) Sanskrit grammar rule base which identifies the Subject, object and Verb from the input sentence.

Three different forms for the sentence "Ram goes to school" are as follows:

रामः विद्यालयम् गच्छति

Raamah Vidyaalayam Gacchati

(Ram) (to school) (goes)

SOV

रामः गच्छति विद्यालयम्

Raamah Gacchati Vidyaalayam

(Ram (goes) (to school)

SVO

गच्छति विद्यालयम् रामः

Gacchati Vidyaalayam Raamah

VOS

In Sanskrit, the best feature is that the position of the word does not tell about the role in the sentence. So the grammatical characteristics (grammar rule base) are used by the system to identify which word plays the role of Subject, Object or Verb and accordingly converts the sentence into SOV format for easy translation. SOV format is used because other Indian languages use the same format for translation. Although there are exceptions to this as in case of interrogative Sanskrit sentence discussed

in the language divergence section earlier. After finalizing the word order, the tokens are generated by using space as the delimiter and forwarded to the next module. If the word is a compound word then Sandhi rule base is applied in reverse to get the desired tokens.

(b) POS Tagging

- The tokens generated from the last phase are processed first by rule base.
- By applying the rules, the tagging is done token by token and forwarded to next phase.
- If no rule is found for any token then the tagged corpus is used to do the tagging with Elasticsearch technique to enhance the processing speed.
- If ambiguity persists then the Sanskrit-UNL dictionary is used to disambiguate the tokens by using UNL attributes and tagging is done accordingly. The tagged tokens are sent to Module-3 for processing.

2. Module-2

This module is the database for the proposed system which consists of:

a) Sanskrit-English bilingual dictionary of more than two lakh words.

b) Sanskrit-UNL dictionary of 17000 words of the general domain [163].

c) Sanskrit tagged corpus of more than four lakhs entries.

As shown in Figure 4.18, this database is used by various modules in the translation process. For enhancing the data access from these dictionaries and tagged dataset, Elasticsearch technique is used which is an open source, scalable, text search and analytical engine. It performs the task of indexing words and

makes them ready to be searched with their position rapidly. It allows the analysis, searching and storing of large volumes of data quickly and in near real time. Searching can be done on any data whether structured or unstructured using this technique.

3. Module-3

   In this module using proposed Sanskrit grammar and CYK parser, the Sanskrit sentence is processed and the parse tree is generated.

4. Module-4

   This module generates the English parse tree equivalent to Sanskrit parse tree. It uses English generation rule base as given in Table 4.4, language divergence rule as discussed in Table 4.3 and bilingual dictionary (Sanskrit-English) for generating the target parse tree. Elasticsearch technique reduces the delay in accessing the data from bilingual dictionaries. Sanskrit-UNL dictionary removes ambiguity among words during target language generation and adds semantic information to the target sentence. Figure 4.19 shows the example of English parse tree generated from equivalent Sanskrit parse tree.

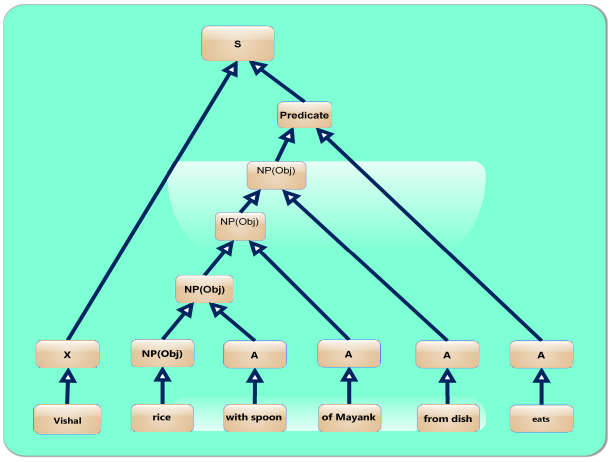

Figure 4.19: English Parse Tree

5. Module-5

This module performs the translation using DMT approach. The bilingual Sanskrit-Hindi dictionary and Sanskrit-UNL dictionary are used to generate TL word for SL word. The word by word replacement is done in this phase. Using module-2 the processing speed of accessing equivalent English word is enhanced. The reordering of word as per target language is done in section 6.

6. Module-6

In module 4, scanning the leaf nodes of the tree from left to right generates the target English sentence. The word order of the Sanskrit language is Subject-Object-Verb whereas for English it is Subject-Verb-Object, so in the final English tree the verb part and the object part are mutually shifted to the desired output. In module 5 for direct approach, reordering of words is done to get the sentence in TL structure . In this phase, the final output is generated and reordering is performed to get the final output.

Chapter 5

Testing and Performance Analysis

After performing design and implementation of the proposed system in previous chapters, this chapter presents testing and performance evaluation of the proposed systems. Various evaluation metrics including BLEU, fluency score and adequacy score are used to evaluate performance of the proposed system using various data sets

5.1 MT Evaluation Methods

In this section, actual evaluation process of the proposed MT systems is depicted. Broadly MT evaluation methods are divided into two categories :

- Traditional Evaluation Methods
- Automatic Evaluation Methods

5.1.1 Traditional Evaluation Methods

This section is highlighting the traditional evaluation methods used for MT evaluation [164].

1. Fluency Test

   Fluency of an MTS gives the measure of amount with which the target text is well-formed according to TL grammar rules. A grammatically well-formed sentence with correct spellings, name and titles which can easily be interpreted

and acceptable by native speaker of the TL is known as the fluent segment [52, 165]. The 4-point scale fluency score proposed by [165] is shown in Table 5.1. This method has been used in the evaluation of Punjabi EnConverter and DeConverter System.

Table 5.1: 4 Point Fluency Score

Fluency Score	4 point Fluency score
1	Incomplete / not Intelligible
2	Acceptable
3	Fair
4	Perfectly Acceptable

2. Adequacy Test

   Adequacy is the measure of an amount of information correctly translated into the TL from SL. It tells about the correctness of translation. Table 5.2 shows four point ranking with corresponding meaning for each rank.

Table 5.2: 4 Point Adequacy Score

Adequacy Score	4 point Adequacy score meaning
1	completely unfaithful
2	less sentence information is conveyed
3	almost complete sentence meaning is conveyed
4	Complete faithful

It has been applied to the evaluation of Hindi-Dogri MTS, Punjabi Deconverter and English-French MT produced by SYSTRAN system.

5.1.2 Automatic Evaluation Method

Bilingual Evaluation Understudy (BLEU) [166]

This method is based on word precision. In this method, adequacy is measured through word precision and for fluency score n-gram precision is used. An exact word to word match is being used in this approach without considering inflections or derivative forms of the words.

5.2 Performance Evaluation of Proposed System

5.2.1 Sanskrit Tagger Performance Evaluation

To test the proposed Sanskrit tagger, data-set DS1 along with suffixes and prefixes has been used. In this work, two architectures Bidirectional Long Short Term Memory (BiLSTM) have been used for POS tagging. For training and testing of these architectures, DS1 is divided into two sections of 80% and 20% respectively. The performance of both architectures is shown in Figure 5.1. The result analysis shows that BiLSTM architecture is beating SLSTM by 0.452% in terms of accuracy.

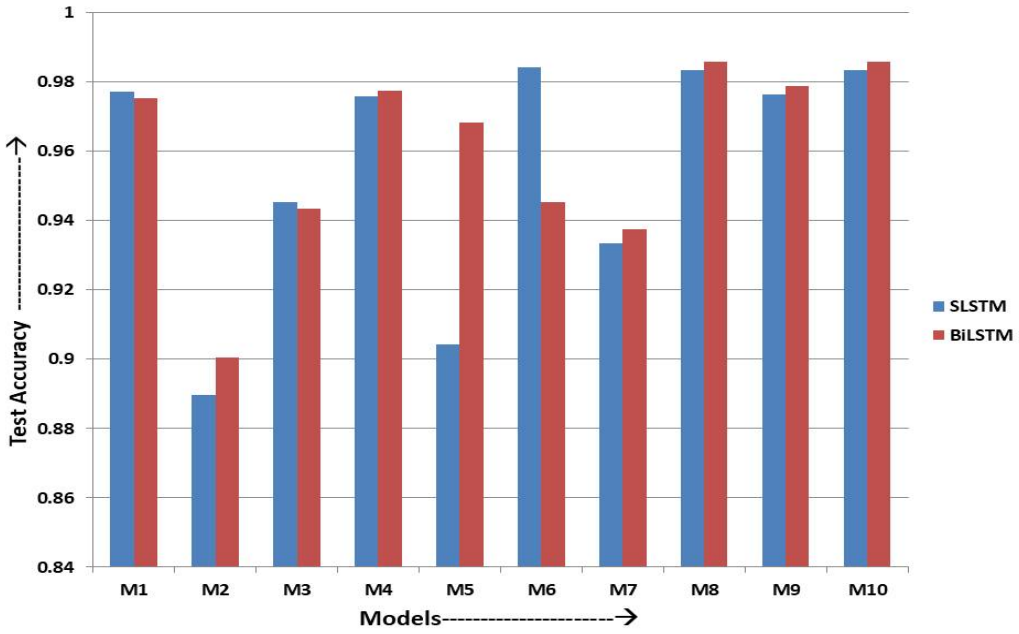

Figure 5.1: Accuracy of SLSTM versus BiLSTM

5.2.2 Evaluation of Proposed Sanskrit to UNL Enconversion System

The proposed system is tested and evaluated on four data-sets (DS2 to DS5) using Fluency score, Adequacy score and BLEU score metrics. Eight evaluators have been requested to test and evaluate the proposed system based on Fluency and Adequacy score metrics. Four evaluators out of eight were expert in Sanskrit language and they are having good knowledge of English language. Other four evaluators were experts in English and were having good knowledge of UNL. Two teams were formed with two members having Sanskrit-English knowledge and other two members having English-UNL knowledge. They were asked to rate the output of the system on a 4-point scale.

1. Fluency Score

   After getting the evaluation scores from experts on DS2, the analysis of the received data is given below:

   - 80% sentences received a score of 4

- 12% sentences received a score of 3
- 4% sentences received a score of 2
- 4% sentences received a score of 1

On the basis of above analysis, performance of proposed system in the form of fluency score achieved is 3.68 out of 4.

The analysis of performance ranking received for DS3 from the evaluators are as follows:

- 73.33% sentences received a score of 4
- 15.17% sentences received a score of 3
- 5.66% sentences received a score of 2
- 5.34% sentences received a score of 1

Based on above ranking scores the fluency score achieved by the proposed system is 3.54 out of 4.

The fluency ranking obtained from evaluators for DS4 are as follows:

- 81.36% sentences received a score of 4
- 14.64% sentences received a score of 3
- 2.53% sentences received a score of 2
- 1.47% sentences received a score of 1

Based on above ranking scores, fluency score achieved by the proposed system is 3.84 out of 4.

The analysis of ranking obtained from evaluators for DS5 are as follows:

- 80.74% sentences received a score of 4
- 13.36% sentences received a score of 3

- 3.55% sentences received a score of 2
- 2.35% sentences received a score of 1

Based on above ranking scores the fluency score achieved by the proposed system is 3.76 out of 4.

2. Adequacy Score

The adequacy score of the proposed has also been calculated for DS2 to DS4. From evaluators the ranking for adequacy score for DS2 has also been received and the analysis is shown below:

- 83.64% sentences received a score of 4
- 10.1% sentences received a score of 3
- 4.6% sentences received a score of 2
- 1.65% sentences received a score of 1

On the basis of above ranking analysis, the Adequacy score obtained for DS2 is 3.75 out of 4.

The analysis of ranking obtained from evaluators for DS3 are as follows:

- 81.33% sentences received a score of 4
- 10.92% sentences received a score of 3
- 5.50% sentences received a score of 2
- 2.25.34% sentences received a score of 1

From above ranking analysis, the Adequacy score for DS3 of the proposed system has been found to be 3.69 out of 4.

The analysis of ranking obtained from evaluators for DS4 are as follows:

- 82.15% sentences received a score of 4
- 14.15% sentences received a score of 3
- 2.40% sentences received a score of 2
- 1.30% sentences received a score of 1

From above ranking analysis, the Adequacy score for DS3 of the proposed system has been found to be 3.85 out of 4.

3. BLEU score

The performance of proposed system is also evaluated using automatic evaluation method BLEU. The BLEU scores obtained for the proposed system is shown in Table 5.3.

Data-set	Number of sentences	BLEU Score	Data Information	Data Source
DS2	50	0.78	Sanskrit-UNL (Hare and Tortoise story)	http://unlweb.net/wiki/corpusn
DS3	300	0.83	(70-carinal 50-ordinal 30-fractional 150-simple sentences)	http://unlweb.net/wiki/corpusn
DS4	50	0.78	General sentences available at Spanish server	http://www.unl.fi.upm.es/english/fr_examples.htm
DS5	500	0.85	simple sentences	[167][168]

Table 5.3: BLEU score based evaluation of the Proposed System

Based on the above evaluation analysis, it has been found that proposed system has successfully resolved 46 UNL relations. Figure 5.2 shows the UNL relations successfully resolved by the proposed system.

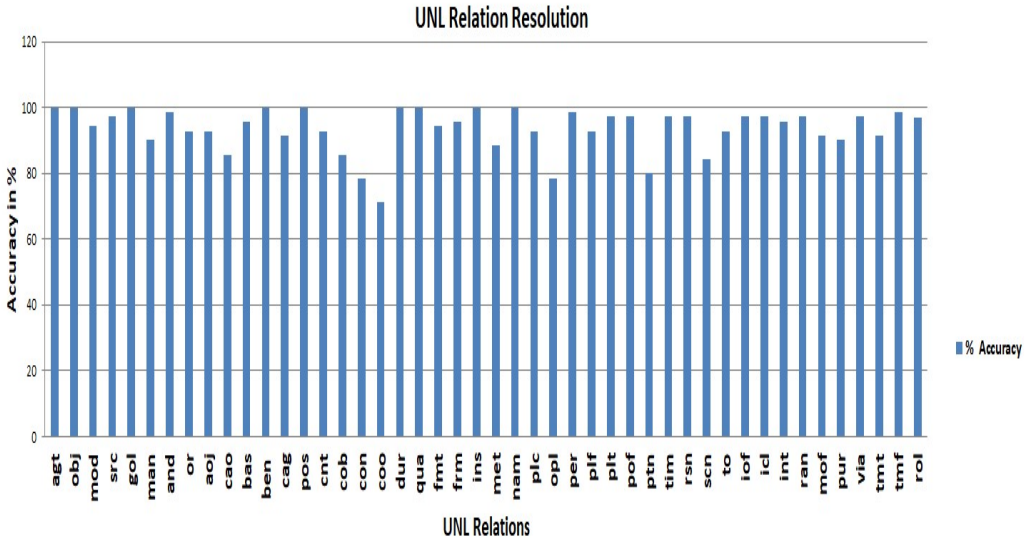

Figure 5.2: UNL Relation Resolution

5.2.3 Evaluation of Proposed Sanskrit to English Translation System

The proposed system has been evaluated using DS5 data-set.

i) Bilingual Evaluation Understudy (BLEU)) [166]

BLEU-2 (2-gram BLEU) score of the proposed system has been found to be 0.7606.

ii) Fluency Score [169] [170]

The proposed system is also evaluated on a 4-point scale fluency score system and achieves a score of 3.63 (out of four). The score indicates degree with which the generated sentence (by the proposed system) obeys the target language grammar rules.

The analysis of the result of 500 sentences is presented in Figure 5.3 and

is explained as follows:

a) 387 sentences achieved score 4 (Perfect Translation)

b) 67 sentences achieved score 3 (Fair Translation)

c) 35 sentences achieved score 2 (Acceptable but require efforts to understand)

d) 11 sentences achieved score 1 (Not acceptable)

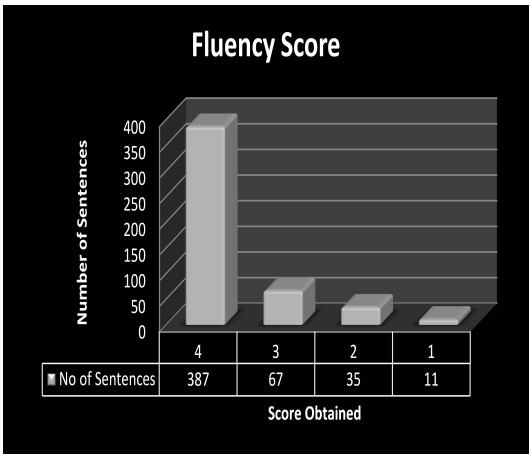

Figure 5.3: Fluency Score

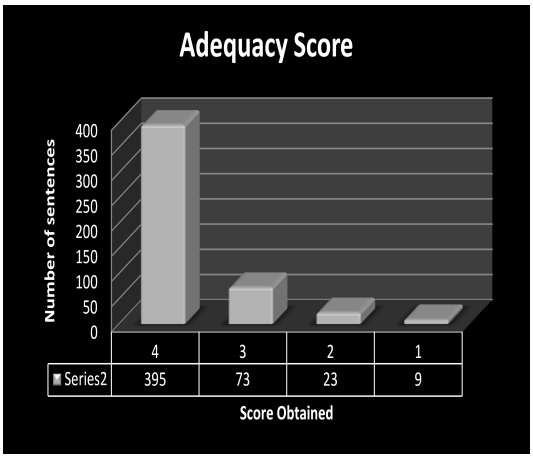

Figure 5.4: Adequacy Score

From the above performance evaluation, it is found that proposed system generates 90.8 (Score 4 and Score 3 sentences) percent grammatically correct sentences.

iii) Adequacy Score

The proposed system obtained 3.72 adequacy score on a 4-scale system. Analysis of DS5 is presented in Figure 5.4 and is explained as follows:

a) 395 sentences achieved score 4 (Complete information transmission)

Table 5.4: Comparison of the Proposed System with Other existing Systems

MT System	BLEU Score	Approach Used	Citation
Hindi-English	0.7502	QNN with RBMT	[105]
Sanskrit-English	NA	RBMT	[75]
Hindi-English	0.2182	CBMT	[93]
Sanskrit-English	0.7606	DMT with RBMT	Proposed System

b) 73 sentences achieved score 3 (Almost complete information transmission)

c) 23 sentences achieved score 2 (Small information transmission)

d) 09 sentences achieved score 1 (No information transmission)

From the above discussion, it has been found that the proposed system transfer 93.6 percent of source information (Score 4 and Score 3 sentences) successfully in the generated sentences.

Table 5.4 is showing comparison of the proposed system with other existing systems and it is found that the proposed system gives better performance in comparison to other existing MT systems. The overall efficiency of the proposed Sanskrit to English MT system is found to be 97.8 percent.

Chapter 6

Conclusions and Future Scope

In this chapter major contributions of the research work done by authors in this thesis are discussed. It also explores the possibilities of future scope of research in the field of machine translation for Sanskrit language. The main focus of this thesis titled "Sanskrit Language Enconversion to Universal Networking Language (UNL)" is to develop a machine translation system for Sanskrit language which converts the Sanskrit text to UNL expressions.

The major research contributions of this thesis are listed below:

Chapter 2 presents a comprehensive literature review of existing MT systems based on various approaches used in MT. The comparison of various MT approaches based on well defined criteria is performed. It also highlights standard MT platforms, linguistic tools and data repositories available for developing new MT systems. Research gaps found in the literature have also been reflected. At last the research objectives of the thesis are listed at the end.

Chapter 3 presents a unique layered architecture of "SANSUNL" system which provides the translation of Sanskrit text to UNL expressions. This chapter includes proposed Sanskrit stemmer, POS tagger, Sanskrit grammar and parse tree generation algorithm.

Chapter 4 presents implementation of the proposed system. It includes Enconversion rule base structure, Sanskrit-Universal Word dictionary structure, five data-sets prepared from different resources. The example based step by step implementation of the

proposed system is also presented. At the end of this chapter, language divergence among Sanskrit and English language with possible recommendations are also presented with the help of Sanskrit to English translation system.

The testing and evaluation of the proposed system is presented in Chapter 5. Both traditional as well as automatic methods are used to test and evaluate the proposed system. The performance of the proposed Sanskrit POS tagger is evaluated using DS1 data-set. The validation is done using golden standard DS2, DS3 and DS4 data-sets. The proposed system is also tested on DS5 data-set. The performance of the proposed system is measured in terms of fluency score, adequacy score and BLEU score. The overall efficiency of the proposed system comes out to be 95% with average fluency score of 3.705 and adequacy score of 3.763. The Sanskrit to English MT system is also tested using DS5 with fluency score of 3.63 and adequacy score of 3.72. The proposed Sanskrit to English MTS has been found to be performing well as compared to existing system in terms of BLEU score.

Thus, the goal of developing machine translation system for Sanskrit to UNL has been successfully attempted in this thesis.

6.1 Future Scope

The development of "SANSUNL" machine translation system and its application to Sanskrit to English machine translation system are viewed as the initial point in this area of research. Thus, there is a wide scope of research in this area. Following are some of the major areas for future extension of proposed work:

- The proposed system can be further extended and validated using more techniques of neural machine translation approach for resolving UNL relations automatically without enconversion rule base.
- The proposed system can be used for translation to other languages using the

DeConverter module.

- There is need to develop Sanskrit deConverter system to generate Sanskrit sentences from UNL expressions.
- There is need for the development of automatic tools which will generate Sanskrit-UNL sentence corpus.

Bibliography

[1] W. J. Hutchins and H. L. Somers, An introduction to machine translation, vol. 362. Academic Press London, 1992.

[2] W. J. Hutchins, ``Machine translation: A brief history,'' Concise history of the language sciences: from the Sumerians to the cognitivists, pp. 431--445, 1995.

[3] N. Kalchbrenner and P. Blunsom, ``Recurrent continuous translation models,'' in Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1700--1709, 2013.

[4] I. Sutskever, O. Vinyals, and Q. V. Le, ``Sequence to sequence learning with neural networks,'' in Advances in neural information processing systems, pp. 3104--3112, 2014.

[5] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, ``On the properties of neural machine translation: Encoder-decoder approaches,'' arXiv preprint arXiv:1409.1259, 2014.

[6] J. Seasly, ``Machine translation: a survey of approaches,'' University of Michigan, Ann Arbor, 2003.

[7] P. Antony, ``Machine translation approaches and survey for indian languages,'' International Journal of Computational Linguistics & Chinese Language Processing, Volume 18, Number 1, March 2013, vol. 18, no. 1, 2013.

[8] V. Fromkin, R. Rodman, and V. Hyams, ``An introduction to language, 9e,'' Boston, MA: Wadsworth, Cengage Learning, 2011.

[9] J. Allen, Natural Language Understanding. Pearson, 2nd ed., 1995.

[10] B. Vauquois, ``A survey of formal grammars and algorithms for recognition and transformation in mechanical translation.,'' in Ifip congress (2), vol. 68, pp. 1114--1122, 1968.

[11] D. C. Swami and P. Saraswati, ``Sanskrit : The mother of all languages,'' 5 2017.

[12] P. Goyal, G. Huet, A. Kulkarni, P. Scharf, and R. Bunker, ``A distributed platform for sanskrit processing,'' Proceedings of COLING 2012, pp. 1011--1028, 2012.

[13] M. Vivker, ``Importance of sanskrit language,'' 9 2013.

[14] P. N., C., ``Relevance of sanskrit in modern age.'' •, 3 2014.

[15] U. IN-Q-TEL, ``Iqt.'' In-Q-Tel Inc, 1999. April.

[16] D. Arnold, Machine translation: an introductory guide. Blackwell Pub, 1994.

[17] A. Gallafent and L. Mullins, ``Machine translation for military.'' Public Radio International, April 2011.

[18] R. Flournoy, ``http://blogs.adobe.com/globalization/2011/06/07/more-content-into-more-languages/.''

[19] H. John, ``Uses and application of machine translation,'' 2009.

[20] G. Randhawa, M. Ferreyra, R. Ahmed, O. Ezzat, and K. Pottie, ``Using machine translation in clinical practice,'' Canadian Family Physician, vol. 59, no. 4, pp. 382--383, 2013.

[21] G. Diana, ``Translation in tourism industry.'' LEXINGTON, January 2017. •.

[22] S. Lappin and H. J. Leass, ``An algorithm for pronominal anaphora resolution,'' Computational linguistics, vol. 20, no. 4, pp. 535--561, 1994.

[23] A. Bharati, V. Chaitanya, R. Sangal, and K. Ramakrishnamacharyulu, Natural language processing: a Paninian perspective. Prentice-Hall of India New Delhi, 1995.

[24] V. Agarwal and P. Kumar, ``Unlization of punjabi text for natural language processing applications,'' Sādhanā, vol. 43, no. 6, p. 87, 2018.

[25] S. Semaan, ``Unlp universal networking language programme,'' TechKnowLogia, pp. 63--65, 1999.

[26] H. Uchida and M. Zhu, ``The universal networking language beyond machine translation,'' in International Symposium on Language in Cyberspace, Seoul, pp. 26--27, 2001.

[27] H. Uchida, M. Zhu, and T. Della Senta, ``Universal networking language,'' UNDL foundation, vol. 2, 2005.

[28] P. Kumar and R. Sharma, ``Punjabi to unl enconversion system,'' Sadhana, vol. 37, no. 2, pp. 299--318, 2012.

[29] ``Unl corpus,'' 2012. Last accessed 04 August 2020.

[30] A. H. Homiedan, ``Machine translation,'' Journal of King Saud University, 1998.

[31] S. Gupta, Natural Language Processing/Speech, NLP and the Web. IIT Bombay, SIC 201, Kanwal Rekhi Building, IIT Bombay, India, edition ed., January 2012.

[32] P. Goyal and R. M. K. Sinha, ``A study towards design of an english to sanskrit machine translation system,'' in Sanskrit Computational Linguistics, pp. 287--305, Springer, 2009.

[33] P. Antony, Computational linguistic tools and machine translation system for Kannada language. PhD thesis, AMRITA VISHWA VIDYAPEETHAM, 2012.

[34] D. Bahdanau, K. Cho, and Y. Bengio, ``Neural machine translation by jointly learning to align and translate,'' arXiv preprint arXiv:1409.0473, 2014.

[35] R. Agrawal, Towards efficient Neural Machine Translation for Indian Languages. PhD thesis, International Institute of Information Technology, Hyderabad, 2017.

[36] V. Narayana, Anusarak: A Device to Overcome the Language Barrier. PhD thesis, Ph. D. thesis, Dept. of CsE, IIT Kanpur, 1994.

[37] G. S. Josan and G. S. Lehal, ``A punjabi to hindi machine translation system,'' in 22nd International Conference on on Computational Linguistics: Demonstration Papers, pp. 157--160, Association for Computational Linguistics, 2008.

[38] V. Goyal and G. S. Lehal, ``Web based hindi to punjabi machine translation system,'' Journal of Emerging Technologies in Web Intelligence, vol. 2, no. 2, pp. 148--151, 2010.

[39] P. Dubey et al., ``Machine translation system for hindi-dogri language pair,'' in Machine Intelligence and Research Advancement (ICMIRA), 2013 International Conference on, pp. 422--425, IEEE, 2013.

[40] T. S. Saini, G. S. Lehal, and V. S. Kalra, ``Shahmukhi to gurmukhi transliteration system,'' in 22nd International Conference on on Computational Linguistics: Demonstration Papers, pp. 177--180, Association for Computational Linguistics, 2008.

[41] T. V. Prasad and G. M. Muthukumaran, ``Telugu to english translation using direct machine translation approach,'' Proceedings of the International Journal of Science and Engineering Investigation, vol. 2, pp. 25--32, 2013.

[42] P. Dubey, ``The hindi to dogri machine translation system: grammatical perspective,'' International Journal of Information Technology, vol. 11, no. 1, pp. 171--182, 2019.

[43] B. J. Dorr, ``Interlingual machine translation a parameterized approach,'' Artificial Intelligence, vol. 63, no. 1, pp. 429--492, 1993.

[44] R. Sinha, K. Ivaraman, A. Agrawal, R. Jain, R. Srivastava, A. Jain, et al., ``Anglabharti: a multilingual machine aided translation project on translation from english to indian languages,'' in Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on, vol. 2, pp. 1609--1614, IEEE, 1995.

[45] R. Sinha and A. Jain, ``Anglahindi: an english to hindi machine-aided translation system,'' MT Summit IX, New Orleans, USA, pp. 494--497, 2003.

[46] S. Dave, J. Parikh, and P. Bhattacharyya, ``Interlingua-based english--hindi machine translation and language divergence,'' Machine Translation, vol. 16, no. 4, pp. 251--304, 2001.

[47] T. Dhanabalan, K. Saravanan, and T. Geetha, ``Tamil to unl enconverter,'' in Proc. Int. Conf. on Universal Knowledge and Language, pp. 1--16, 2002.

[48] T. Dhanabalan and T. Geetha, ``Unl deconverter for tamil,'' in International Conference on the Convergences of Knowledge, Culture, Language and Information Technologies, 2003.

[49] R. M. K. Sinha, ``Integrating cat and mt in anglabharti-ii architecture,'' in 10th EAMT conference, pp. 235--244, 2005.

[50] K. Dey, P. Bhattacharyya, et al., ``Universal networking language based analysis and generation of bengali case structure constructs,'' Res. Comput. Sci, vol. 12, pp. 215--229, 2005.

[51] R. M. K. Sinha and A. Thakur, ``Machine translation of bi-lingual hindi-english (hinglish) text,'' 10th Machine Translation summit (MT Summit X), Phuket, Thailand, pp. 149--156, 2005.

[52] S. Singh, M. Dalal, V. Vachhani, P. Bhattacharyya, and O. P. Damani, ``Hindi generation from interlingua (unl),'' Machine Translation Summit XI, 2007.

[53] M. Jain and O. P. Damani, ``English to unl (interlingua) enconversion,'' in Proc. Second Conference on Language and Technology,(CLT), 2009.

[54] M. Ali, N. Yousuf, S. Ripon, and S. M. Allayear, ``Unl based bangla natural text conversion-predicate preserving parser approach,'' arXiv preprint arXiv:1206.0381, 2012.

[55] P. Kumar and R. K. Sharma, ``Punjabi deconverter for generating punjabi from universal networking language,'' Journal of Zhejiang University SCIENCE C, vol. 14, no. 3, pp. 179--196, 2013.

[56] B. Nair, R. Rajeev, and E. Sherly, ``Language dependent features for unl-malayalam deconversion,'' International Journal of Computer Applications, vol. 975, p. 8887, 2014.

[57] R. Sridhar, P. Sethuraman, and K. Krishnakumar, ``English to tamil machine translation system using universal networking language,'' Sādhanā, vol. 41, no. 6, pp. 607--620, 2016.

[58] H. Darbari, ``Computer-assisted translation system--an indian perspective,'' Machine Translation Summit VII, 13th-17th September, pp. 80--85, 1999.

[59] K. Vijayanand, S. I. Choudhury, and P. Ratna, ``Vaasaanubaada: automatic machine translation of bilingual bengali-assamese news texts,'' in Language Engineering Conference, 2002. Proceedings, pp. 183--188, IEEE, 2002.

[60] N. Ata, B. Jawaid, and A. Kamaran, ``Rule based english to urdu machine translation,'' in Conference on Language and Technology, 2007.

[61] A. Choudhary and M. Singh, ``Gb theory based hindi to english translation system,'' in Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on, pp. 293--297, IEEE, 2009.

[62] A. Bharati and A. Kulkarni, ``Anusaaraka: An accessor cum machine translator,'' Department of Sanskrit Studies, University of Hyderabad, Hyderabad, pp. 1--75, November 2009.

[63] M. Bhadra, S. K. Singh, S. Kumar, M. Agrawal, R. Chandrasekhar, S. K. Mishra, G. N. Jha, et al., ``Sanskrit analysis system (sas),'' in Sanskrit Computational Linguistics, pp. 116--133, Springer, 2009.

[64] K. K. Batra and G. Lehal, ``Rule based machine translation of noun phrases from punjabi to english,'' International Journal of Computer Science Issues, vol. 7, no. 5, pp. 409--413, 2010.

[65] M. Barkade and P. R. DEVALE, ``English to sanskrit machine translation semantic mapper,'' International Journal of Engineering Science and Technology, vol. 2, no. 10, pp. 5313--5318, 2010.

[66] G. Pathak and S. Godse, ``English to sanskrit machine translation using transfer approach,'' in International Conference on Methods and Models in Science and Technology, pp. 122--126, Pune: American Institute of Physics, 2010.

[67] A. Kumar, V. Mittal, and A. Kulkarni, ``Sanskrit compound processor,'' in Sanskrit Computational Linguistics, pp. 57--69, Springer, 2010.

[68] A. Kulkarni, S. Pokar, and D. Shukl, ``Designing a constraint based parser for sanskrit,'' in Sanskrit Computational Linguistics, pp. 70--90, Springer, 2010.

[69] K. K. BATRA and G. LEHAL, ``Automatic translation system from punjabi to english for simple sentences in legal domain,'' INTERNATIONAL JOURNAL OF TRANSLATION, vol. 23, no. 1, 2011.

[70] M. Gopal and G. N. Jha, ``Tagging sanskrit corpus using bis pos tagset,'' in Information Systems for Indian Languages, pp. 191--194, Springer, 2011.

[71] P. Bahadur, A. Jain, and D. Chauhan, ``Etrans-a complete framework for english to sanskrit machine translation,'' in International Journal of Advanced Computer Science and Applications (IJACSA) from International Conference and workshop on Emerging Trends in Technology, pp. 52--59, Citeseer, 2012.

[72] G. B. Kumar and K. N. Murthy, ``Ucsg shallow parser,'' in Computational Linguistics and Intelligent Text Processing, pp. 156--167, Springer, 2006.

[73] V. K. Gupta, N. Tapaswi, and S. Jain, ``Knowledge representation of grammatical constructs of sanskrit language using rule based sanskrit language to english language machine translation,'' in Advances in Technology and Engineering (ICATE), 2013 International Conference on, pp. 1--5, IEEE, 2013.

[74] P. Desai, A. Sangodkar, and O. P. Damani, ``A domain-restricted, rule based, english-hindi machine translation system based on dependency parsing,'' in Proceedings of the 11th International Conference on Natural Language Processing, pp. 177--185, 2014.

[75] P. Upadhyay, U. C. Jaiswal, and K. Ashish, ``Transish: Translator from sanskrit to english-a rule based machine translation,'' International Journal of Current Engineering and Technology E-ISSN, pp. 2277--4106, 2014.

[76] M. V. Reddy and M. Hanumanthappa, ``Indic language machine translation tool: English to kannada/telugu,'' in Multimedia Processing, Communication and Computing Applications, pp. 35--49, Springer, 2013.

[77] A. Kulkarni, ``A deterministic dependency parser with dynamic programming for sanskrit,'' in Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013), pp. 157--166, 2013.

[78] R. Sinha, ``Research and projects.'' http://www.cse.iitk.ac.in/users/rmk/proj/proj.html.

[79] Sitender and S. Bawa, ``Survey of indian machine translation systems,'' IJCST(Internati onal Journal of Computer Sci ci enc e And Tec hnology), vol. 3, pp. 286--290, March-June 2012.

[80] S. K. Dwivedi and P. P. Sukhadeve, ``Machine translation system in indian perspectives,'' Journal of computer science, vol. 6, no. 10, p. 1111, 2010.

[81] R. Jain, R. Sinha, and A. Jain, ``Anubharti-using hybrid example-based approach for machine translation,'' STRANS-2001, IIT Kanpur, pp. 20--32, 2001.

[82] G. Garje and G. Kharate, ``Survey of machine translation systems in india,'' International Journal on Natural Language Computing (IJNLC), vol. 2, no. 4, pp. 47--67, 2013.

[83] S. Naskar and S. Bandyopadhyay, ``Use of machine translation in india: Current status,'' AAMT Journal, pp. 25--31, 2005.

[84] I. I. R. Lab, ``English-hindi machine translation system.''

[85] ``Saskrit heritage resources.''

[86] V. Mishra and R. Mishra, ``Study of example based english to sanskrit machine translation,'' Journal of Research and Development in Comp Sc. And Engg.(37)(January-June 2008), 2008.

[87] R. Udupa and T. A. Faruquie, ``An english-hindi statistical machine translation system,'' in Natural Language Processing--IJCNLP 2004, pp. 254--262, Springer, 2005.

[88] F. OCH, ``Google translator,'' in Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, (Prague), pp. 858--867, Association for Computational Linguistics, 2007.

[89] V. Goyal and G. S. Lehal, ``Evaluation of hindi to punjabi machine translation system,'' arXiv preprint arXiv:0910.1868, 2009.

[90] N. Sharma, English to Hindi Statistical Machine Translation System. PhD thesis, THAPAR UNIVERSITY PATIALA, 2011.

[91] N. Khan, A. Waqas, U. Bajwa, and N. Durrani, ``English to urdu hierarchical phrase-based statistical machine translation,'' in WSSANLP2013, (Japan), pp. 72--76, October 2013.

[92] A. Ali, A. Hussain, and M. K. Malik, ``Model for english-urdu statistical machine translation,'' World Applied Sciences, vol. 24, pp. 1362--1367, 2013.

[93] K. Sachdeva, R. Srivastava, S. Jain, and D. M. Sharma, ``Hindi to english machine translation: Using effective selection in multi-model smt.,'' in LREC, pp. 1807--1811, 2014.

[94] P. Dungarwal, R. Chatterjee, A. Mishra, A. Kunchukuttan, R. Shah, and P. Bhattacharyya, ``The iit bombay hindi - english translation system at wmt 2014,'' ACL 2014, p. 90, 2014.

[95] A. L. Lagarda, D. Ortiz-Martínez, V. Alabau, and F. Casacuberta, ``Translating without in-domain corpus: Machine translation post-editing with online learning techniques,'' Computer Speech & Language, vol. 32, no. 1, pp. 109--134, 2015.

[96] B. Jawaid, A. Kamran, and O. Bojar, ``English to urdu statistical machine translation: Establishing a baseline,'' in Proceedings of the Fifth Workshop on South and Southeast Asian Natural Language Processing, pp. 37--42, 2014.

[97] M. T. Singh, R. Borgohain, and S. Gohain, ``An english-assamese machine translation system,'' International Journal of Computer Applications, vol. 93, no. 4, 2014.

[98] G. K. Saha, ``The eb-anubad translator: A hybrid scheme,'' Journal of Zhejiang University Science A, vol. 6, no. 10, pp. 1047--1050, 2005.

[99] R. Ananthakrishnan, M. Kavitha, J. H. Jayprasad, R. S. Chandra Shekhar, and S. M. Sawani Bade, ``Matra: A practical approach to fully-automatic indicative english-hindi machine translation,'' in Symposium on Modeling and Shallow Parsing of Indian Languages (MSPIL'06), 2006.

[100] M. Christopher and U. M. Rao, ``Il-ilmt sampark: A hybrid machine translation system,'' in 32nd All India Conference of Linguistics (AICL32), pp. 69--75, Lucknow University, Lucknow, December 2010.

[101] G. S. Josan and G. S. Lehal, ``A punjabi to hindi machine transliteration system,'' Computational Linguistics and Chinese Language Processing, vol. 15, no. 2, pp. 77--102, 2010.

[102] V. Mishra and R. Mishra, ``Approach of english to sanskrit machine translation based on case-based reasoning, artificial neural networks and translation rules,'' International Journal of Knowledge Engineering and Soft Data Paradigms, vol. 2, no. 4, pp. 328--348, 2010.

[103] A. Shahnawaz and R. Mishra, ``Translation rules and ann based model for english to urdu machine translation,'' INFOCOMP Journal of Computer Science, vol. 10, no. 3, pp. 25--35, 2011.

[104] V. Goyal and G. S. Lehal, ``Hindi to punjabi machine translation system,'' in Proceedings of the 49th Annual Meeting of the Association for Computational

Linguistics: Human Language Technologies: Systems Demonstrations, pp. 1--6, Association for Computational Linguistics, 2011.

[105] R. Narayan, V. Singh, and S. Chakraverty, ``Quantum neural network based machine translator for hindi to english,'' The Scientific World Journal, vol. 2014, 2014.

[106] Microsoft, ``Microsoft translator accelerates use of neural networks across its offerings.'' https://blogs.msdn.microsoft.com/translation/2017 /11/15/microsoft-translator-accelerates-use-of-neural -networks-across-its-offerings/, November 2017.

[107] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., ``Google's neural machine translation system: Bridging the gap between human and machine translation,'' arXiv preprint arXiv:1609.08144, 2016.

[108] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, ``Convolutional sequence to sequence learning,'' arXiv preprint arXiv:1705.03122, 2017.

[109] J. Gehring, M. Auli, D. Grangier, and Y. N. Dauphin, ``A convolutional encoder model for neural machine translation,'' arXiv preprint arXiv:1611.02344, 2016.

[110] F. Faes, ``Amazon and lionbridge share stage to market neural machine translation.'' https://slator.com/technology/amazon-and-lionbridge-share-stage-to-market-neural-machine-translation/, April 2018.

[111] A. Globalization and A. Bi-Weekly, ``History and frontier of the neural machine translation compared to smt, nmt can train multiple features jointly and does not need prior domain knowledge, enabling zero-shot translation. in addition to higher bleu score and better sentence structure, nmt can also help reduce

morphology errors, syntax errors, and word order errors of smt.," Synced AI TECHNOLOGY and INDUSTRY REVIEW, 2017.

[112] S. Wang, W. Zhou, and C. Jiang, ``A survey of word embeddings based on deep learning," Computing, vol. 102, no. 3, pp. 717--740, 2020.

[113] S. K. Knapp, ``Accelerate fpga macros with one-hot approach," Electronic Design, vol. 38, no. 17, pp. 71--78, 1990.

[114] T. Mikolov, K. Chen, G. Corrado, and J. Dean, ``Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[115] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, ``Distributed representations of words and phrases and their compositionality," in Advances in neural information processing systems, pp. 3111--3119, 2013.

[116] J. Pennington, R. Socher, and C. Manning, ``Glove: Global vectors for word representation," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), (Doha, Qatar), pp. 1532--1543, Association for Computational Linguistics, Oct. 2014.

[117] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, ``Bag of tricks for efficient text classification," arXiv preprint arXiv:1607.01759, 2016.

[118] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, ``Deep contextualized word representations," arXiv preprint arXiv:1802.05365, 2018.

[119] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, ``Attention is all you need," in Advances in neural information processing systems, pp. 5998--6008, 2017.

[120] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ``Bert: Pre-training of deep bidirectional transformers for language understanding,'' arXiv preprint arXiv:1810.04805, 2018.

[121] X. Zheng, H. Chen, and T. Xu, ``Deep learning for chinese word segmentation and pos tagging,'' in Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 647--657, 2013.

[122] H. T. Ng and J. K. Low, ``Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based?,'' in Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pp. 277--284, 2004.

[123] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, ``A unified tagging solution: Bidirectional lstm recurrent neural network with word embedding,'' arXiv preprint arXiv:1511.00215, 2015.

[124] B. Plank, A. Søgaard, and Y. Goldberg, ``Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss,'' arXiv preprint arXiv:1604.05529, 2016.

[125] M. Labeau, K. Löser, and A. Allauzen, ``Non-lexical neural architecture for fine-grained pos tagging,'' in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 232--237, 2015.

[126] M. Ballesteros, C. Dyer, and N. A. Smith, ``Improved transition-based parsing by modeling characters instead of words with lstms,'' arXiv preprint arXiv:1508.00657, 2015.

[127] W. Ling, T. Luís, L. Marujo, R. F. Astudillo, S. Amir, C. Dyer, A. W. Black, and I. Trancoso, ``Finding function in form: Compositional character models for open vocabulary word representation,'' arXiv preprint arXiv:1508.02096, 2015.

[128] W. Ling, I. Trancoso, C. Dyer, and A. W. Black, ``Character-based neural machine translation,'' arXiv preprint arXiv:1511.04586, 2015.

[129] M. R. Costa-Jussa and J. A. Fonollosa, ``Character-based neural machine translation,'' arXiv preprint arXiv:1603.00810, 2016.

[130] M. L. Forcada, M. Ginestí-Rosell, J. Nordfalk, J. O'Regan, S. Ortiz-Rojas, J. A. Pérez-Ortiz, F. Sánchez-Martínez, G. Ramírez-Sánchez, and F. M. Tyers, ``Apertium: a free/open-source platform for rule-based machine translation,'' Machine translation, vol. 25, no. 2, pp. 127--144, 2011.

[131] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush, ``Opennmt: Open-source toolkit for neural machine translation,'' arXiv preprint arXiv:1701.02810, 2017.

[132] M.-T. Luong and C. D. Manning, ``Stanford neural machine translation systems for spoken language domains,'' in Proceedings of the International Workshop on Spoken Language Translation, pp. 76--79, 2015.

[133] Microsoft, ``Microsoft translator launching neural network based translations for all its speech languages.'' https://blogs.msdn.microsoft.com/translation/2016/11/15/microsoft-translator-launching-neural-network-based-translations-for-all-its-speech-languages/, November 2016.

[134] Yandex, ``Yandex blog.'' https://yandex.com/company/blog/one-model-is-better-than-two-yu-yandex-translate-launches-a-hybrid-machine-translation-system/, September 2017.

[135] P. Koehn, ``Moses--statistical machine translation system,'' 2009.

[136] A. B. Phillips, ``Cunei: open-source machine translation with relevance-based models of each translation instance,'' Machine Translation, vol. 25, no. 2, p. 161, 2011.

[137] M. Post, Y. Cao, and G. Kumar, ``Joshua 6: A phrase-based and hierarchical statistical machine translation system,'' The Prague Bulletin of Mathematical Linguistics, vol. 104, no. 1, pp. 5--16, 2015.

[138] J. Devlin, R. Zbib, Z. Huang, T. Lamar, R. Schwartz, and J. Makhoul, ``Fast and robust neural network joint models for statistical machine translation,'' in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, pp. 1370--1380, 2014.

[139] M. Federico, N. Bertoldi, and M. Cettolo, ``Irstlm: an open source toolkit for handling large scale language models,'' in Ninth Annual Conference of the International Speech Communication Association, 2008.

[140] R. Rosenfeld and P. Clarkson, ``Cmu-cambridge statistical language modeling toolkit v2,'' 1997.

[141] A. Stolcke, ``Srilm-an extensible language modeling toolkit,'' in Seventh international conference on spoken language processing, 2002.

[142] A. Vaswani, Y. Zhao, V. Fossum, and D. Chiang, ``Decoding with large-scale neural language models improves translation,'' in Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1387--1392, 2013.

[143] I. Hyderabad, ``Machine translation and natural language processing lab.'' http://ltrc.iiit.ac.in/, 04 2018.

[144] J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov, and E. Marsi, ``Maltparser: A language-independent system for data-driven

dependency parsing," Natural Language Engineering, vol. 13, no. 2, pp. 95--135, 2007.

[145] C. Pune, ``Indian language technology proliferation and development centre," 04 2018.

[146] P. Baker, A. Hardie, T. McEnery, H. Cunningham, and R. J. Gaizauskas, ``Emille, a 67-million word corpus of indic languages: Data collection, mark-up and harmonisation.," in LREC, 2002.

[147] G. N. Jha, ``The tdil program and the indian langauge corpora intitiative (ilci).," in LREC, 2010.

[148] J. Tiedemann, ``News from opus-a collection of multilingual parallel corpora with tools and interfaces," in Recent advances in natural language processing, vol. 5, pp. 237--248, 2009.

[149] learnsanskrit.org, ``Sanscript," 2020. Last accessed 29 July 2020.

[150] M. Gopal, D. Mishra, and D. P. Singh, ``Evaluating tagsets for sanskrit," in Sanskrit Computational Linguistics, pp. 150--161, Springer, 2010.

[151] M. Gopal and G. N. Jha, ``Indian language part of speech tagger (il-post)," 2007. http://sanskrit.jnu.ac.in/corpora/tagset.jsp.

[152] R. Chandershekhar and G. N. Jha, Part-of-Speech Tagging for Sanskrit. PhD thesis, Special Centre for Sanskrit Studies, JNU Delhi, http://sanskrit.jnu.ac.in/corpora/JNU-Sanskrit-Tagset.htm, 2007.

[153] S. Sarkar and S. Bandyopadhyay, ``Design of a rule-based stemmer for natural language text in bengali," in Proceedings of the IJCNLP-08 workshop on NLP for Less Privileged Languages, 2008.

[154] A. Kulkarni, ``A sanskrit computational toolkit,'' 2020. Last accessed 29 July 2020.

[155] W. Zhang, T. Du, and J. Wang, ``Deep learning over multi-field categorical data,'' in European conference on information retrieval, pp. 45--57, Springer, 2016.

[156] J.-C. Chappelier, M. Rajman, et al., ``A generalized cyk algorithm for parsing stochastic cfg.,'' TAPD, vol. 98, no. 133-137, p. 5, 1998.

[157] D. H. Younger, ``Recognition and parsing of context-free languages in time n3,'' Information and control, vol. 10, no. 2, pp. 189--208, 1967.

[158] T. Li and D. Alagappan, ``A comparison of cyk and earley parsing algorithms,'' icar.cnr.it, 2006.

[159] X. Chen, L. Xu, Z. Liu, M. Sun, and H. Luan, ``Joint learning of character and word embeddings,'' in Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015.

[160] B. J. Dorr, ``Machine translation divergences: A formal description and proposed solution,'' Computational Linguistics, vol. 20, no. 4, pp. 597--633, 1994.

[161] P. Goyal and R. M. K. Sinha, ``Translation divergence in english-sanskrit-hindi language pairs,'' in Sanskrit Computational Linguistics, pp. 134--143, Springer, 2008.

[162] V. Mishra and R. Mishra, ``Divergence patterns between english and sanskrit machine translation,'' INFOCOMP, vol. 8, no. 3, pp. 62--71, 2009.

[163] Sitender and S. Bawa, ``Sansunl: A sanskrit to unl enconverter system,'' IETE Journal of Research, vol. 0, no. 0, pp. 1--12, 2018.

[164] G. Van Slype, ``Critical study of methods for evaluating the quality of machine translation,'' Prepared for the Commission of European Communities Directorate General Scientific and Technical Information and Information Management. Report BR, vol. 19142, 1979.

[165] V. Goyal, Development of a Hindi to Punjabi Machine Translation system. PhD thesis, Punjabi University, Patiala, 2010.

[166] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, ``Bleu: a method for automatic evaluation of machine translation,'' in Proceedings of the 40th annual meeting on association for computational linguistics, pp. 311--318, Association for Computational Linguistics, 2002.

[167] P. Gupt, Prayogik Sanskrit Vayakaran class 10. Lakshmi Publications, 2010.

[168] R. Narale, Sanskrit for English Sspeaking People. Prabhat Parkashan, 2010.

[169] LDC, ``Linguistic data annotation specification:assesment of adequacy and fluency in translations. revision 1.5.,'' tech. rep., Linguistic Data Consortium, 2005.

[170] P. Kumar and R. Sharma, UNL Based Machine Translation System for Punjabi Language. PhD thesis, Thapar University, 2012.

List of Publications

1. Sitender and Seema Bawa, "SANSUNL : A Sanskrit to UNL Enconverter System", IETE Journal of Research (2018):1-12. doi.org/10.1080/03772063.2018.1528187 (SCIE- Journal, Impact Factor 2.333).

2. Sitender and Seema Bawa, "Sanskrit to English Machine Translation using hybridization of Direct and Rule Based approach". Neural Computing and Applications(2020). doi.org/10.1007/s00521-020-05156-3.(SCI-Journal, Impact Factor=5.606).

3. Sitender and Seema Bawa, "Survey of Indian Machine Translation Systems". International Journal of Computer Science and Technology (2012) 286-290.

4. Sitender and Seema Bawa. "Sanskrit to Universal Networking Language Enconverter System based on Deep learning and Context Free Grammar". Multimedia Systems (SCI Journal, Impact Factor=1.935).

5. Sitender, Seema Bawa, Munish Kumar and Sangeeta. "A comprehensive survey on machine translation for English, Hindi and Sanskrit languages". Journal of Ambient Intelligence and Human Computing(2021). DOI https://doi.org/10.1007/s12652-021-03479-0(SCI Journal, Impact Factor=7.10)

Appendix A

Handling of Logical or Syntactic Errors

The proposed Sanskrit grammar has been used to check the syntactic or logical error if any in the input sentence.

The proposed Context Free Grammar in Chomsky Normal Form has been used and removes any bad formed sentence due to syntactical or logical error. Also the CYK parser is capable enough to remove any ambiguity in the input sentence and reports if any in the parsing stage before entering to the UNL expression generation phase.