ANN and Rule Based Model for English to Sanskrit Machine Translation

Vimal Mishra[1]
R.B.Mishra[2]
[1] Research Scholar, Department of Computer Engineering, Institute of Technology, Banaras Hindu University, (IT-BHU), Varanasi-221005, U.P., India, vimal.mishra.upte@gmail.com
[2] Professor, Department of Computer Engineering, Institute of Technology, Banaras Hindu University, (IT-BHU), Varanasi-221005, U.P., India, ravibm@bhu.ac.in

Abstract. The development of Machine Translation system for ancient language such as Sanskrit language is much more fascinating and challenging task. Due to lack of linguistic community, there are no wide work accomplish in Sanskrit translation while it is mother language by virtue of its importance in cultural heritage of India. In this paper, we integrate a traditional rule based approach of machine translation with Artificial Neural Network (ANN) model which translates an English sentence (source language sentence) into equivalent Sanskrit sentence (target language sentence). We use feed forward ANN for the selection of Sanskrit word like noun, verb, object, adjective etc from English to Sanskrit User Data Vector (UDV). Due to morphological richness of Sanskrit language, this system makes limited use of syntax and uses only morphological markings to identify Subject, Object, Verb, Preposition, Adjective, Adverb and as well as Conjunctive sentences also. It uses limited parsing for part of speech (POS) tagging, identification of clause, its Subject, Object, Verb etc and Gender-Number-Person (GNP) of noun, adjective and object. This system represents the translation between the SVO and SOV classes of languages. This system gives translation result in GUI form and handles English sentences of different classes.

Keywords: Sanskrit, Machine Translation, ANN, Rule Based Model, English to Sanskrit Machine Translation, ANN and Rule Based Model for Machine Translation.

(Received September 08, 2009 / Accepted May 07, 2010)

1. Introduction

India is a multilingual country with eighteen constitutionally recognized languages [23]. Even though, Sanskrit is understand by 0.01% (49,736) as per census of India, 1991. Therefore, machine translation provides a solution in breaking the language barrier within the country. Correct karaka assignment poses the greatest problem in this regard [22] [2]. There have been many MT systems for English to other foreign languages as well as to Indian languages but none for English to Sanskrit machine translation. Sanskrit language is the mother of all Indian languages. Mostly, all Indian languages are originating from Sanskrit language. So, if we develop English to Sanskrit machine translation system then it will easier to translate from English to any Sanskrit originating Indian languages. Some works on Sanskrit parser and morphological analyzers have done earlier which is briefly described below.

The work of Ramanujan, P. [21] describes that morphological analysis of Sanskrit is the basic requirement for the computer processing of Sanskrit. The Nyaya (Logic), Vyakarana (Grammar) and Mimamsa (Vedic interpretation) is a suitable solution that covers syntactic, semantic and contextual analysis of Sanskrit sentence. P. Ramanujan has developed a Sanskrit parser 'DESIKA', which is Paninian grammar based analysis program. DESIKA includes Vedic processing and shabda-bodha as well. In DESIKA, these are separate modules for the three functions of the system: generation, analysis and reference. Generation of nominal or verbal class of word is carried out by the user specifying the word and the applicable rules being activated. In analysis, the syntactic identification and assignment of functional roles for every word is carried out using the Karaka-Vibhakti mappings. In the reference module, a complete 'trace' of the process of generation or analysis is planned to be provided, besides information.

In [3], Rick Briggs uses semantic nets (knowledge representation scheme) to analyze sentences unambiguously. He compares the similarity between English to Sanskrit and provides the theoretical implications of their equivalence.

In [4], Huet has developed a Grammatical Analyzer System, which tags NPs (Noun Phrase) by analyzing sandhi, samasa and sup affixation.

The works in Sanskrit processing tools and Sanskrit authoring system have carried out Jawaharlal Nehru University, New Delhi-India. It is currently engaged in karaka Analyzer, sandhi splitter and analyzer, verb analyzer, NP gender agreement, POS tagging of Sanskrit, online Multilingual amarakosa, online Mahabharata indexing and a model of Sanskrit Analysis System (SAS)[6].

Morphological analyzers for Sanskrit have been developed by Akshara Bharathi Group at Indian Institute of Technology, Kanpur-India and University of Hyderabad-India.

We have developed a prototype model of English to Sanskrit machine translation (EST) system using ANN model and rule based approach. ANN model gives matching of equivalent Sanskrit word of English word which handles noun and verb. The ANN based system gives us faster matching of English noun (subject or object) or verb to appropriate Sanskrit noun (subject or object) or dhaatu. The rule based model generates Sanskrit translation of the given input English sentence using rules that generate verb and noun for Sanskrit. The rule based approaches mostly make use of hand written transfer rules to the translation of substructures from source language (English sentence) to target language (Sanskrit sentence). The main advantages of rule based approaches are easy implementation and small memory requirement [5].

We have divided our work into the following sections. Section 2 presents machine translation model with the ANN based system that describes encoding, input-output generation and decoding of User Data Vector (UDV) with the help of table and algorithm; the detail description of the system overview with the help of information flow diagram and algorithm. Section 3 presents implementation and results of the translation in GUI form. Section 4 presents evaluations using table and column chart. The conclusions and scope for future work are mentioned in Section 5.

2. Machine Translation Model

We are giving below the machine translation model for English to Sanskrit translation system which is integrated system of ANN model and rule based model. ANN model gives matching of equivalent Sanskrit word of English word which handles noun and verb, while rule based model generate Sanskrit translation of the given input English sentence using rules that generate verb and noun for Sanskrit.

2.1. ANN Based System

In the ANN based model, we use feed forward ANN for the selection of equivalent Sanskrit word such as noun (subject or object) and verb of English sentence. In feed forward ANN, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. The use of feed forward ANN overcomes the output vector limitation [24]. Our motivation behind use of feed forward ANN in language processing tasks are work of Nemec, Peter [19] and Khalilov, Maxim et al. [8].

We basically perform three steps in ANN based system which are given below.

1. Encoding of User Data Vector (UDV)
2. Input-Output generation of UDV
3. Decoding of UDV

2.1.1 Encoding of UDV

English alphabet consists of twenty-six characters which can be represented by five bit binary (25 =32, it ranges from 00000 to 11111). First, we write alphabet (a-z) into five bit binary in which alphabet "a" as 00001, to avoid the problem of divide by zero and alphabet "z" as 11010. For the training into ANN system, we make the alphabet to decimal coded form which is obtained by dividing each to thirty-two. This gives us input word in decimal coded form and output in corresponding Sanskrit word in roman script. The encoding of UDV rules are given below in algorithm I.

Algorithm I

Step 1 Represent the English alphabet by 5 bit binary, such as assigning "a" as 00001 and "z" as 11010.

Step 2 Divide each values of alphabet by 25 (32) and convert them into decimal coded form.

Step 3 Make separate UDV for noun (subject or object) and verb of English and corresponding noun (subject or object) and verb of Sanskrit as output.

Step 4 Input of each UDV has five data values.

In the Encoding of UDV, we provide the data values to the noun (subject or object) and verb. In case of UDV for verb, UDV have five data values, say x1, x2, x3, x4 and x5 for input in English and six data values, say y1, y2, y3, y4 ,y5 and y6 for output in Sanskrit. In case of UDV for noun, UDV have five data values, say x1, x2, x3, x4 and x5 for input in English and seven data values, say y1, y2, y3, y4 ,y5, y6 and y7 for output in Sanskrit.

2.1.2 Input-Output Generation of UDV

We prepare the input-output pair of data for the two to five characters verb and noun in English as input and corresponding verb and noun in Sanskrit as output. We prepare UDV of noun (subject or object) and verb which is of maximum of five characters. In case of two characters noun or verb etc, we add three dummy values which range between 0.007 and 0.009, as suffix to UDV to make them five characters noun or verb. Similarly, we add two and one dummy values for three and four characters noun or verb etc respectively to make them five characters noun or verb. After preparing the UDV, we train the UDV through feed forward ANN and then test the UDV. We get the output of Sanskrit word in the UDV form. The name of our data sets have called UDV here, which is used in feed forward ANN for the selection of equivalent Sanskrit word such as noun (subject or object) and verb of English sentence [11] [12] [13].

2.1.3 Decoding of UDV

The output given by ANN model is in decimal coded form. Each values of a data set is compared with the decimal coded values of alphabet, one by one and the values with minimum difference is taken with its corresponding alphabet.

2.2 A Rule Based Model

Figure 1 shows the information flow in EST system using flowchart. The description of the main module of EST system is given below.

2.2.1 Sentence Tokenizer Module

The sentence tokenizer module split the English sentences into tokens (words) using split method of string tokenizer class in Java. The outputs of the sentence tokenizer module are given to POS Tagger module.

2.2.2 POS Tagger Module

Part–of-Speech tagging is the process of assigning a part–of-speech (such as a noun, verb, pronoun, preposition, adverb and adjective) to each word in a sentence. In POS Tagger module, the Part–of-Speech (POS) tagging is done on each word in the input English sentence. This part–of-speech tagging method falls under the rule-based (linguistic) category. The rule-based taggers use hand coded rules to assign tags to words. The output of POS tagger module is given to rule base engine.

2.2.3 Rule Base Engine

The rule base engine consists of a set of rules that is written for extraction of *karataa*, root *dhaatu*, object, adverb, adjective, preposition etc. for this purposes, rule base engine uses ANN based system for the selection of equivalent of Sanskrit word such as subject, object, verb, adjective, preposition etc of English sentence.

2.2.4 Root Dhaatu Extraction Module

In Sanskrit, verb form is generated according to *dhaatu* which depend upon number and person of noun. The equivalent of Sanskrit *dhaatu* is taken from the output of ANN based system which is given to rule base engine to generate the verb form in Sanskrit. The algorithmic steps are given below in algorithm II.

Algorithm II

Step 1 Create a database that has three rows as person and three columns as number.

Step 2 Store dhaatu in all fields of database file.

Step 3 Check the tense of the sentence.

Step 4 If tense of sentence is present tense then create lat lakar in Sanskrit.

Step 5 If step 4 is yes then add suffix "ati", "atih" and "anti" at case[1,1], case[1,2] and case[1,3] respectively.

Step 6 Add suffix "asi", "athh" and "ath" at case [2, 1], case [2, 2] and case [2, 3] respectively.

Step 7 Add suffix "aami", "aavah" and "aamah" at case [3, 1], case [3, 2] and case [3, 3] respectively.

Similarly, we have created all types of tenses (lakar) which depend upon number and person of noun.

2.5 Generation of bhvaadigana Dhaatu

The generation algorithm of bhvaadigana Dhaatu is shown below in table I.

Table I: Suffix Addition in bhvaadigana Dhaatu: Present Tense

P	Present tense			Imperative		
	S	D	Pl	S	D	Pl
1	ati	atah	anti	atu	ataam	antu
2	asi	athah	atha	a	atam	ata
3	aami	aavah	aamah	aani	aava	aama

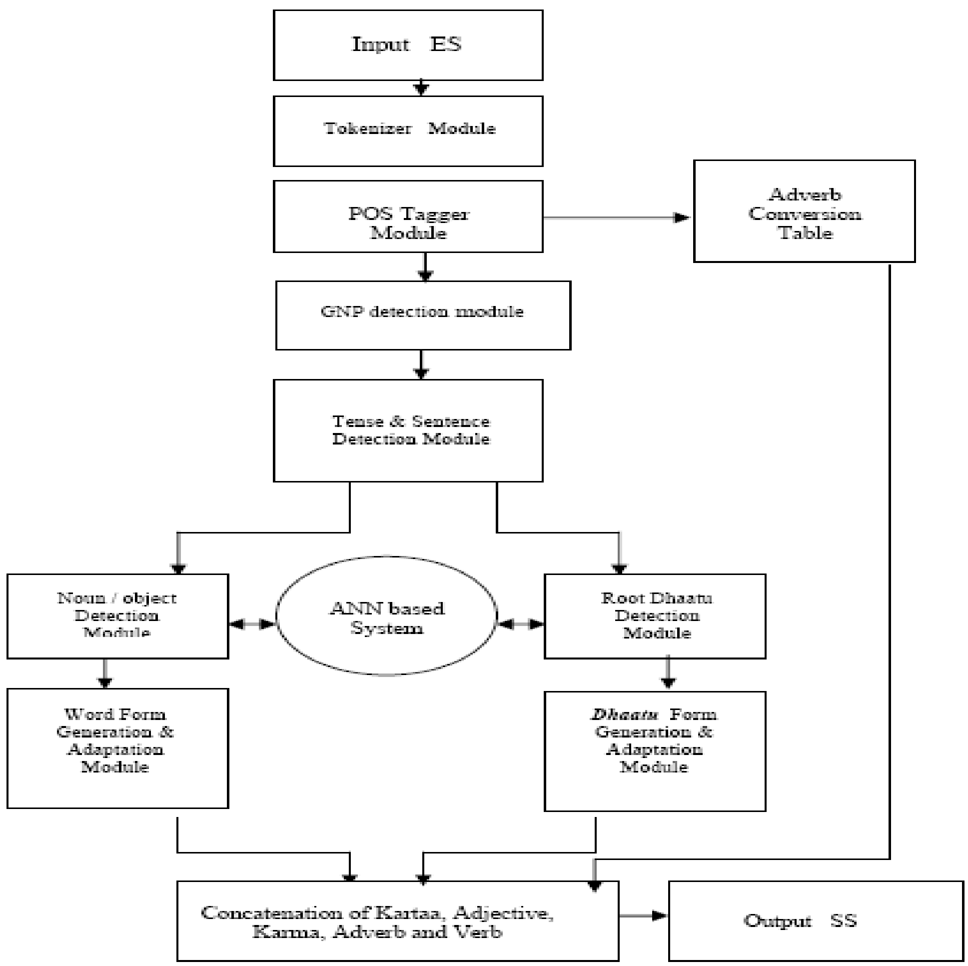

Figure2. Information Flow in EST Model Model

2.6 Word Extraction Module

The word extraction module deals kartaa and object. Here, we denote "word" for kartaa and object both. First, we take the equivalent of Sanskrit word for *kartaa* and object from the output of ANN system which is given to rule base engine to generate the word in Sanskrit.

A stepwise algorithm to create word form in Sanskrit is given below.

Algorithm III

Step 1 Check the suffix of the word.

Step 2 If suffix of Sanskrit word has come with “a”, “aa”, “i”, “ii”, “u”, “uu”, “ri”, “o” and “au” then create case in number.

Step 3 Check the gender of the word: masculine, feminine and neuter then generate corresponding case.

Step 4 Check the number and person of the word.

Step 5 Check the case in Sanskrit that depends upon preposition used in English sentence.

2.2.7 Declension of Nouns and Adjectives

The word is recognized by its suffix and we prepare a database which has different word type with respect to gender of the word. The nouns and adjectives are created using word extraction module.

2.2.8 Adverb Conversion Table

Adverb is that word which does not depend upon the gender, person and number (GNP) of noun. So, it is called *avayava* (indeclinable) in Sanskrit. We take commonly used adverb in English and corresponding in Sanskrit, using data table. Sanskrit has three Person: First (1), Second (2) and third (3). Sanskrit has three Number: Singular (S), Dual (D) and Plural (Pl) [7] [9] [18].

2.2.9 Generation of akAranta Word

A word that ends with character “a” is called akAranta word. For example, the akAranta word is “raama”. The generation algorithm of akAranta word is given below.

Algorithm IV

Step 1 If number is singular and vibhakti is Nominative (N) then add “h” as suffix.

Step 2 If number is singular and vibhakti is Accusative (A) then add “am” as suffix.

Step 3 If number is singular and vibhakti is Instrumental (I) then add “en” as suffix.

Step 4 If number is singular and vibhakti is Dative (D) then add “aay” as suffix.

Step 5 If number is singular and vibhakti is Ablative (Ab) then add “aat” as suffix.

Step 6 If number is singular and vibhakti is Genitive (G) then add “asya” as suffix.

Step 7 If number is singular and vibhakti is Locative (L) then add “e” as suffix.

Step 8 If number is dual then add “au” as suffix at vibhakti Nominative, Vocative (V) and Accusative.

Step 9 If number is dual then add “abhyaam” as suffix at vibhakti Instrumental, Dative and Ablative.

Step 10 If number is dual then add “ayoh” as suffix at vibhakti Genitive and Locative.

Step 11 If number is plural then add “ah” as suffix at vibhakti Nominative and Vocative.

Step 12 If number is plural and vibhakti is Accusative then add “an” as suffix.

Step 13 If number is plural and vibhakti is Instrumental then add “aih” as suffix.

Step 14 If number is plural then add “ebhyah” as suffix at vibhakti Dative and Ablative.

Step 15 If number is plural and vibhakti is Genitive then add “anaam” as suffix.

Step 16 If number is plural and vibhakti is Locative then add “esu” as suffix.

The generation algorithm of akAranta word is shown as below in table II.

Table II: Suffix Addition in akAranta Word: Masculine and Neuter Gender

Vibhakti	Masculine word			Neuter word		
	S	D	Pl	S	D	Pl
N	h	au	ah	m	e	aani
V	-	au	ah	-	e	aani
A	am	au	an	m	e	aani
I	en	abhyaam	aih	en	abhyaam	aih
D	aay	abhyaam	ebhyah	aay	abhyaam	ebhyah
Ab	aat	abhyaam	ebhyah	aat	abhyaam	ebhyah
G	asya	ayoh	anaam	asya	ayoh	anaam
L	e	ayoh	esu	e	ayoh	esu

Word set of akAranta : Masculine = [raama, nara, bhakta, mayura, baala, putra, janaka, nripa, sura,

chandra, surya, shisya, khaga, loka, dharma, anala, anila, khala, upahaara, vaanara, gaja, vansha, pika].
Word set of akAranta: Neuter = [phala, vana, pustaka, jala, gagana, patra, mitra, kamala, kusuma, mukha, pushpa].

Similarly, we have created all types of word with all gender.

3. Implementation and Results

Our EST system has been implemented on window platform using Java. The ANN model is implemented using MATLAB 7.1 neural network tool. We use feed forward ANN that gives matching of equivalent Sanskrit word of English word which handles noun and verb. We have a data set of 125 input-output pair for verb. The input, hidden and output values for verb is taken 5, 38 and 6. The training is terminated at a training error of 10^{-3} after 300 epochs. For the noun, we have 100 input-output pair in which the input, hidden and output values are taken 5, 15 and 7. This training is terminated at a training error of 10^{-2} after 300 epochs. The Run-time convergences for verb and noun are shown in figure 2 and figure 3, respectively. We obtain the intermediate result of the error convergences for the training and testing cases of ANN based model for the noun and verb. The final result is obtained by concatenating of the Sanskrit word for the noun and verb correspondingly by ANN model.

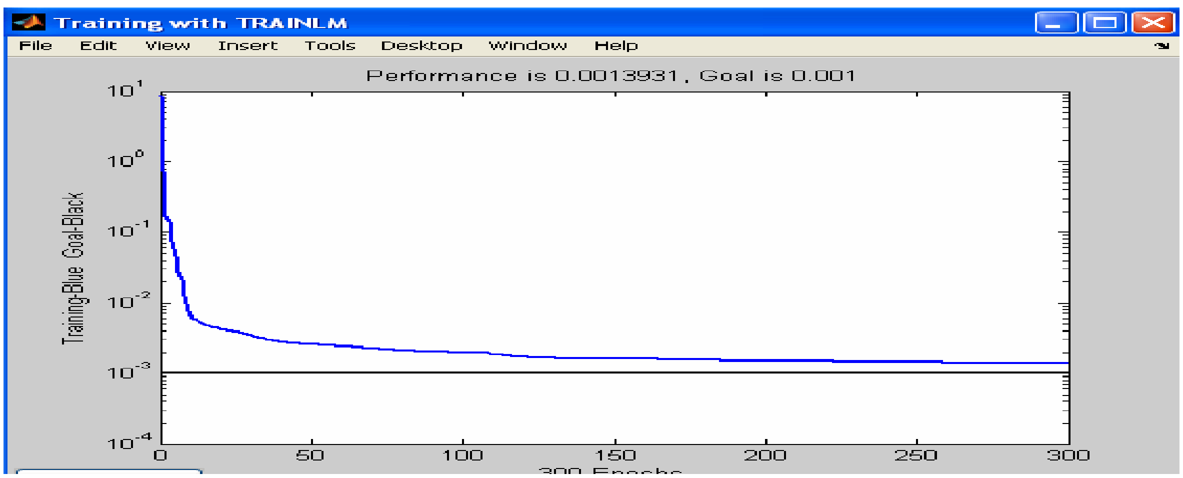

Figure 2: Run-time Convergences for Verb

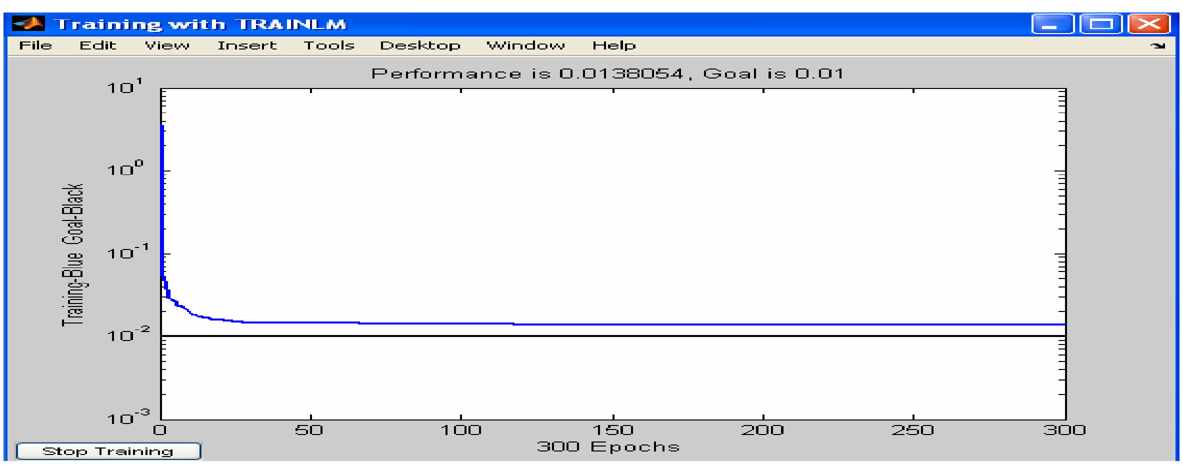

Figure 3: Run-time Convergences for Noun

Finally, by the rule based model, the Sanskrit words for subject, object, verb, adverb, adjective and conjunction depending on their corresponding word in English are concatenated. Our EST system handles English sentences of [SV], [SVO], [SVO, Adverb], [SVO, Adverb], [SVO, Adjective], [SVO, Preposition], [SVO, Interrogative], [SVO, Adjective, Negative], [SVO, Adjective], [SVO, Conjugative] and [SVO, Complex] class. The results for the English sentences (ES) and their corresponding translation in Sanskrit sentence (SS) that is obtained from our EST system with [SVO, Adverb] class and [SVO, Conjugative] class are shown in figure 4 and 5, respectively.

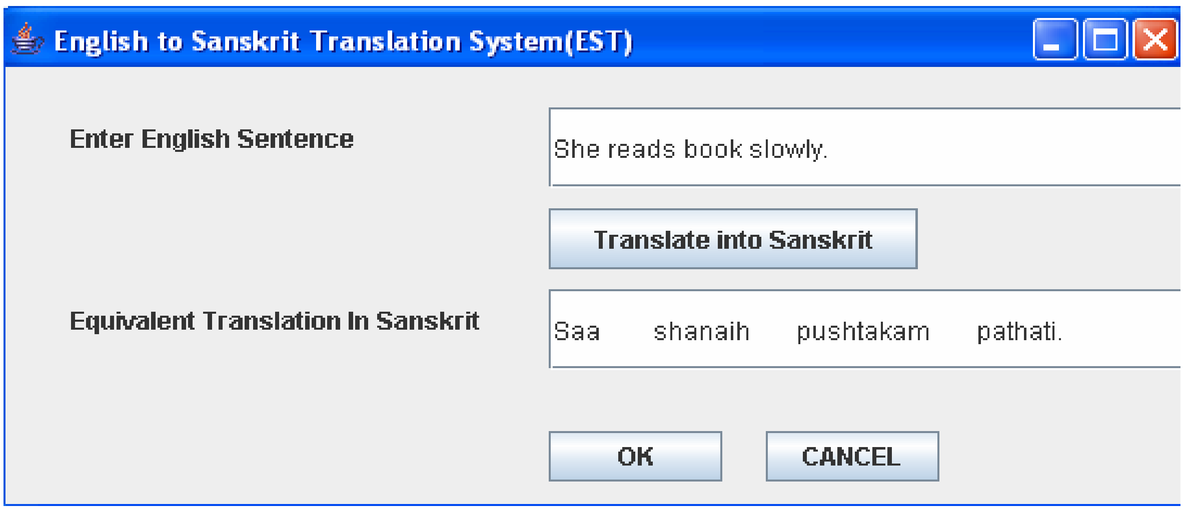

Figure 4: A Sample Output from EST System: [SVO, Adverb] Class of ES

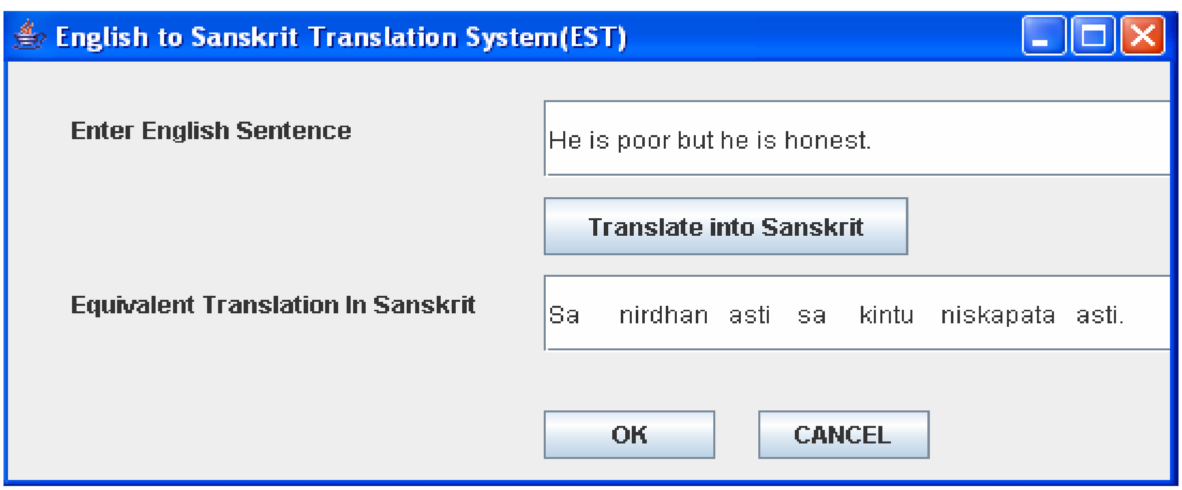

Figure 5: A Sample Output from EST System: [SVO, Conjugative] Class of ES

First, English sentence is split up into tokens and tokens are matched with Sanskrit word using ANN. We check the gender, number and person (GNP) of the noun, adjective and preposition. According to suffixes of the words, we generate corresponding word form. Then, verb form is generated that depends upon the number and person of the noun. We arrange subject, adverb adjective, preposition and verb, in order to obtain the desired translation into Sanskrit language.

4. Evaluations

We evaluate the performance of our EST system [14] [15] that handle the different classes of English sentences using different MT evaluation methods such as BLEU (BiLingual Evaluation Understudy) [20], unigram Precision (P), unigram Recall (R), F-measure (F) [10] and METEOR (M) [1] score. The evaluation scores of our EST system are encouraging which are calculated among randomly twenty selected sentences (ES) with our EST system (C) including reference translations (R) [16] [17] that are given below.

(1) ES: I am fond of eating mango.
C: *Aham aamram bhuktvaa preman asmi.*
R: *Aham aamram gajadhvaa preman asmi.*

(2) ES: Having gone to town, Ram drinks water.
C: *Ramah gramam gatvaa jalam pibati.*
R: *Ramah gramam itvaa jalam pibati.*

(3) ES: I like reading book.
C: *Aham pathitum pustakam vanaami.*
R: *Aham pathitum pustakam ichchati.*

(4) ES: After finishing his work, the boy went to school.
C: *Baalakah svakaasyam kritvaa vidyaalayam agacchat.*
R: *Baalah svakaasyam kritvaa vidyaalayam agacchat.*

(5) ES: On seeing his friend, he was very pleased.
C: *Sah svamitram drishtvaa ateeva aprasiidat.*
R: *Sah svamitram niriisya ateeva aprasiidat.*

(6) ES: I called him, but he gave me no answer.
C: *Aham tasya avadataam kintu sah mayaa uttasam na avadat.*
R: *Aham tasya avadataam parantu sah mayaa uttasam na avadat.*

(7)ES: He tried hard, but he did not succeed.
C: *Sah kathin prayaasam akarot kintu sah saphalam na abhavat.*
R: *Sah kathin prayaasam akarot parantu sah saphalam na abhavat.*

(8) ES: The town where I grew up is the Uttar Pradesh.
C: *Gramam kutra aham ajaaye Uttar Pradesh asti.*
R: *Nagaram kutra aham ajaaye Uttar Pradesh asti.*

(9) ES: You will always regret the boy whemn you did this.
C: *Tyam sada divasam setsyaase kadaa tvam ta akaroh.*
R: *Tyam sada divasam setsyaase kadaa tvam ayam akaroh.*

(10) ES: I know the place which you mention.
C: *Aham sthaanam janaami ya tvam aharudaath.*
R: *Aham sthaanam janaami ya tvam ankitah karoshi.*

(11) ES: He reads Sanskrit quickly.
C: *Sah Sanskrit jhatiti pathati.*
R: *Sah Sanskrit drutam pathati.*

(12) ES: She said of laughing.
C: *Saa hasitvaa abraviit.*
R: *Saa vihasya abraviit.*

(13) ES: Ram will attend this meeting.
C: *Ramah asyaam sabhaayaam anuvartishyate.*
R: *Ramah asyaam samaagamayaam anuvartishyate.*

(14) ES: Ram married Sita.
C: *Ramah Sitaaya sah paanigsahanam akrot.*
R: *Ramah Sitaaya sah vivaaham akrot.*

(15) ES: Ram is afraid of lion.
C: *Ramah singhaat vibheti.*
R: *Ramah sheraat vibheti.*

(16) ES: Ram is regular uses of library.
C: *Ramah pustakaalayasya aharvisham prayogam karoti.*
R: *Ramah pustakaalayasya aharvisham upyogam karoti.*

(17) ES: Ram is feeling hungry.
C: *Ramen ksuthitaa anubhuuyate.*
R: *Ramen bhuukhaa anubhuuyate.*

(18) ES: This gutter smells foul.
C: *Asmaat jalanirgamaat malinam jighrati.*
R: *Asmaat jalanirgamaat durgandham jighrati.*

(19) ES: Either Ram or Krishna had come here.
C: *Ramah Krishnah vaa atra aagacchet.*
R: *Ramah Krishnah athavaa atra aagacchet.*

(20) ES: He is poor but he is honest.
C: *Sah nirdhana asti sa kintu niskapata asti.*
R: *Sah nirdhana asti sa parantu niskapata asti.*

The evaluation scores for randomly selected twenty English sentences of various classes are shown in table III.

Table III: The Evaluation Scores for Randomly Selected Twenty English Sentences of Various Classes

S	BLEU	P	R	F	M
1	0.325	0.8	0.8	0.8	0.811
2	0.325	0.8	0.8	0.8	0.811
3	0.48	0.75	0.75	0.75	0.7516
4	0.6717	0.8	0.8	0.8	0.811
5	0.325	0.8	0.8	0.8	0.811
6	0.6359	0.888	0.888	0.888	0.8888
7	0.6358	0.888	0.888	0.888	0.8888
8	0.8101	0.8571	0.714	0.633	0.642
9	0.714	0.875	0.875	0.875	0.876
10	0.5953	0.8333	0.7143	0.7247	0.7256
11	0.27	0.75	0.75	0.75	0.7516
12	0.1667	0.6667	0.6667	0.6667	0.7082
13	0.27	0.75	0.75	0.75	1.0
14	0.325	0.8	0.8	0.8	0.8057
15	0.1667	0.6667	0.6667	0.6667	0.7082
16	0.4075	0.8	0.8	0.8	0.8075
17	0.1667	0.6667	0.6667	0.6667	0.7082
18	0.2708	0.75	0.75	0.75	0.7827
19	0.3050	0.80	0.80	0.80	0.8112
20	0.5435	0.8571	0.8571	0.8571	0.8596

The comparative score of different MT evaluation methods such as BLEU (BiLingual Evaluation Understudy), unigram Precision (P), unigram Recall (R), F-measure (F) and METEOR (M) for randomly twenty selected English sentences of various classes are shown in figure 6.

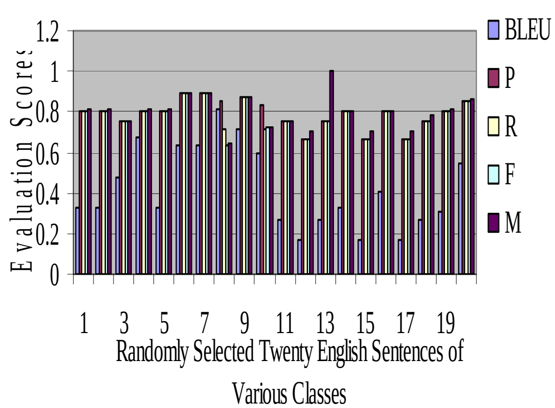

Figure 6: Evaluation Scores for Randomly Selected Twenty English Sentences of Various Classes

5. Conclusions and Scope for Future Work

The present work is an attempt to translate English sentence into equivalent Sanskrit sentence. Our system handles English sentences of types: (i) simple subject, object and verb; (ii) subject, object, adverb and verb; (iii) subject, object, adjective and verb; (iv) subject, object, preposition and verb; (v) interrogative sentences and (vi) compound sentences. The difficulty and shortcoming of this integrated method is that we have taken the output of ANN system manually. We will interfaced ANN output into our Java program, in future to avoid typo mistake. We will integrate this model with CBR (Case Based Reasoning) and data mining approach in future.

References

[1] Banerjee, S. and Alon Lavie, '*METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments*', In Proceedings of Workshop on "Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization", ACL 2005, pages 65-73, Ann Arbor, Michigan, 2005.

[2] Bharati, Akshar and Kulkarni, Amba, '*Sanskrit and computational linguistics*', First International Sanskrit Computational Symposium, Hyderabad, pp 1-12, 2007.

[3] Briggs, Rick, '*Knowledge Representation in Sanskrit and Artificial Intelligence*', The AI Magazine, pp 33–39, 1985.

[4] Huet, G'erard, '*Towards Computational Processing of Sanskrit, Recent Advances in Natural Language Processing*', In Proceedings of the International Conference ICON, Mysore, India, 2003.

[5] Jain, Renu, Sinha, R.M.K. and Ajai Jain, '*ANUBHARTI: Using Hybrid Example-Based Approach for Machine Translation*', In Proceedings Symposium on Translation Support Systems (STRANS2001), Kanpur, India, February 15-17, 2001.

[6] Jha, Girish N et al., '*Towards a Computational analysis system for Sanskrit*', In Proc. of first National symposium on Modeling and Shallow

parsing of Indian Languages at Indian Institute of Technology Bombay, pp 25-34, 2006.

[7] Kale, M. R. '*A Higher Sanskrit Grammar'*, 4th Ed, Motilal Banarasidas Publishers Pvt. Ltd., 2005.

[8] Khalilov, Maxim et al., '*Neural Network Language Models for Translation with Limited Data'*, In Proceedings of 20th IEEE International Conference on Tools with Artificial, pp 445-451, 2008.

[9] Macdonnel, Arthur A. '*A Sanskrit Grammar for Students'*, 3rd Ed, Motilal Banarasidas Publishers Pvt. Ltd., 2003.

[10] Melamed, R. Green, and J. Turian, '*Precision and Recall of Machine Translation'*, In Proceedings of the HLTNAACL 2003, Short Paper, pp 61–63, Edmonton, Canada, 2003.

[11] Mishra, Vimal and Mishra, R. B., '*English to Sanskrit Machine Translation System: A Hybrid Model'*, In Proceedings of International Joint Conference on Information and Communication Technology (IJCICT 2010), 9th-10th January, pp 174-180, IIMT, Bhubaneswar, India, 2010.

[12] Mishra, Vimal and Mishra, R. B., '*Handling of Infinitives in English to Sanskrit Machine Translation*', International Journal of Artificial Life Research (IJALR), IGI Publication, USA, 2010 (Accepted).

[13] Mishra, Vimal and Mishra, R. B., '*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 (IJKESDP), InderScience Publication, UK, 2010 (To be published in Vol.2, No.4).

[14] Mishra, Vimal and Mishra, R. B., '*Evaluation of English to Sanskrit machine translation*', Journal "ATTI Della Fondazione GIORGIO RONCHI", Vol. LXIV, No. 3, pp 345-353, 2009.

[15] Mishra, Vimal and Mishra, R. B., '*Performance Evaluation of English to Sanskrit Machine Translation System: A Novel Approach*', In Proceedings of International Joint Conference on Information and Communication Technology (IJCICT 2010), 9th-10th January, pp 181-186, IIMT, Bhubaneswar, India, 2010.

[16] Mishra, Vimal and Mishra, R. B., 'Study of Example based English to Sanskrit Machine Translation', Journal of Research and Development in Computer Science and Engineering, Polibits, Vol. 37, pp 43-54, 2008.

[17] Mishra, Vimal and Mishra, R. B., *Divergence handling in English to Sanskrit machine translation*, International Journal of Information and Communication Technology , InderScience Publication, UK, 2009 (Accepted).

[18] Nautiyal, Chakradhar, '*Vrihad Anuvaad Chandrika*', 4th Ed, Motilal Banarasidas Publishers Pvt. Ltd., 1997.

[19] Nemec, Peter, '*Application of Artificial Neural Networks in Morphological Tagging of Czech*', pp 1-8, 2003.

[20] Papineni, Kishore & Roukos, Salim et al., '*BLEU: A Method for Automatic Evaluation of Machine Translation*', In Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics, pp. 1– 9, 2002.

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

[22] Samantaray, S.D., '*Example based machine translation approach for Indian languages'*, In Proceedings of ICCS, pp 1-10, 2004.

[23] Sinha, R.M.K. and Jain, Ajai, '*AnglaHindi: An English to Hindi Machine Translation System*', MT Summit IX, New Orleans, USA, Sept. 23-27, 2003.

[24] Thrun, S.B., '*The MONK's Problems: a performance comparison of different learning algorithms*' Technical Report CMU-CS-91-197, Carnegie Mellon University, 1991.