



Rule Based Machine Translation for Assamese-English Using Apertium

Pranjal Das¹, Kalyanee Kanchan Baruah², Abdul Hannan³, Shikhar Kr. Sarma⁴

Department of Information Technology
IST, Gauhati University
Guwahati
India

Abstract: Machine Translation is the process of using a software application to convert a source language into a target language without the intervention of human. It is an important part of Natural Language Processing. Here we have proposed a method for translating Assamese to English sentences and vice versa using a Rule Based Machine Translation (RBMT) approach. RBMT uses linguistic rules to analyze the inputted content in the source language to generate the output text in the target language. We have used Apertium toolkit, which is a free and open-source platform for RBMT. We have used two monolingual dictionaries and a bilingual dictionary for building the Assamese-English RBMT system. We wrote some transfer rules to deal with word-reordering and other grammatical stuffs.

Keywords: Apertium, Transfer Rules, Assamese, Morphology, Machine Translation, Dictionary

I. Introduction

In this paper, we have described the development of the Assamese-English (asm-eng) language pair using Apertium RBMT. Assamese is a language mainly spoken by the people of Assam, a state in the north-east of India [9]. In Assam, not much has been done in the field of NLP and Machine Translation. Assamese is a computationally underdeveloped language and research on Natural Language Processing (NLP) is still at its developing stage. Today, almost everything is web-based and majority of the information is available in English. So the purpose of our project is to enable the common Assamese people, who have lesser knowledge about the English language, to stay abreast with the recent advancements of the world by providing him an Machine Translation interface. The purpose of the project is also to let the world know about the Assamese language. Since Indian languages are morphologically very rich and agglutinative in nature, majority of the MT projects in India are hybrid or statistical based. But we believe that if we have a firm grip on the language, RBMT can be better. In this paper, we have briefly described about the design of the RBMT system using the Apertium platform. Then, we have described about the dictionaries and the rules used. Following this, we have given an evaluation of the quality of output of our system and finally we concluded our paper with a discussion and prospective for future work.

II. Related Works

Machine Translation in India is not a new process. There are many machine translation projects in India [2]. Some of the Machine Translation systems for Indian languages are:

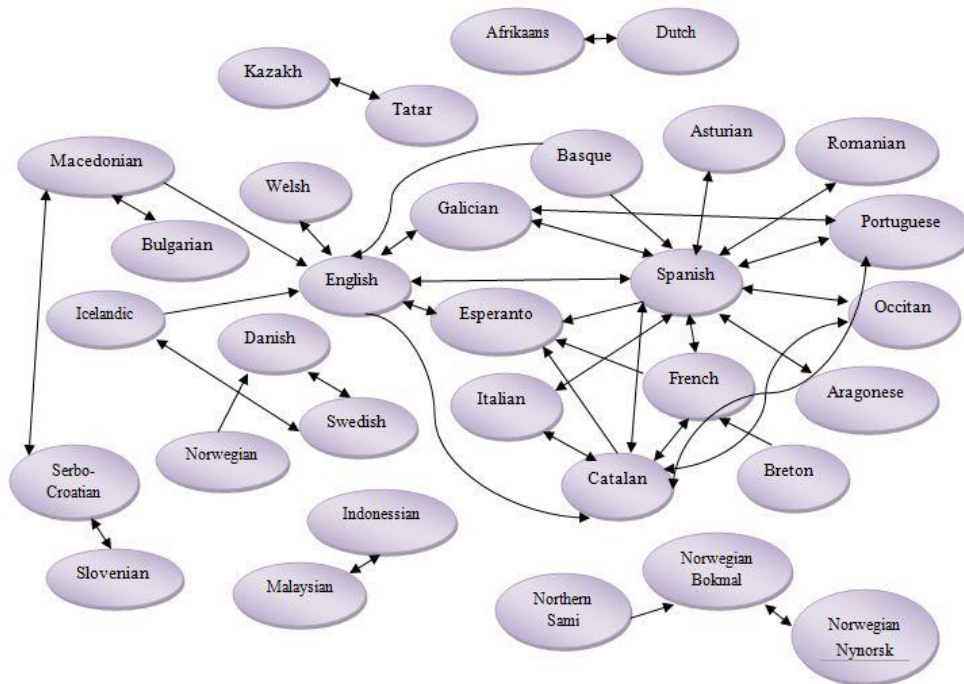
Table I Some Machine Translation Projects in India

Machine Translation System	Source-Target Language	Developer	Approach Used
ANGLABHARATI (1991)	English-Indian Languages (primarily Hindi)	IIT, Kanpur	Pseudo-interlingua
UCSG-based MT	English-Kannada	University of Hyderabad	Transfer based
Anusaaraka (1995)	English to Hindi and five other Indian Languages to Hindi	IIT, Kanpur	LWG Mapping/PG
VAASAANUBAADA (2002)	Bengali- Assamese	Kommaluri Vijayanand, S Choudhury and Pranab Ratna	EBMT
Anuvadakh	English to Hindi, Urdu, Oriya, Bangla, Marathi, Tamil	EILMT consortium	Hybrid Approach

Several universities in India, along with CDAC have been working on improving Machine Translation for Indian Languages. MHRD, India is also taking several measures to make Machine Translation serve for the people of India.

We are using Apertium to develop our RBMT system. There are many language pairs in use for Apertium. Work is still going on for some language pairs. Some of the stable language pairs that are currently in use are shown in the following figure:

Figure 1: Stable language pairs of Apertium.



All the released pairs are stored in the *trunk* [3] of Apertium.

III. Files and Tools used for Implementation

A. Apertium

Apertium is a free and open-source platform used in Rule Based Machine Translation. Apertium uses FST (finite-state transducers) for lexical processing to treat many kinds of multi-word expressions. Hidden Markov models (HMM) are used in Apertium for part-of-speech tagging. Apertium also uses chunking, which is finite-state based used for structural transfer.

B. Lttoolbox

It is a finite state toolkit used for lexical processing, morphological analysis and generation of words [4]. Three programs are split in the Lttoolbox:

lt-comp – the compiler

lt-proc – the processor

lt-expand – generates all possible mappings in the dictionary between surface forms and lexical forms.

C. Monolingual Dictionaries

The monolingual dictionary or the morphological dictionary contains the rules how words in that dictionary are inflected. It is written in XML format. The monolingual dictionary establishes the correspondences between Surface Forms (SFs) and Lexical Forms (LFs) [5]. For our language pair Assamese-English, we have two monolingual dictionaries –

apertium-asm-eng.asm.dix – for Assamese

apertium-asm-eng.eng.dix – for English

D. Bilingual Dictionary

The bilingual dictionary contains relations between words and symbols in the two languages i.e. it establishes the correspondences between source language LFs and target language LFs [5]. For our language pair, we have the following bilingual dictionary –

apertium-asm-eng.asm-eng.dix

E. Transfer Rules

The transfer rule file contains some grammatical rules for how a source language will be changed to a target language. These rules are mainly used because all languages do not follow the same word order. For e.g. English follows SVO construction, while Assamese follows SOV word order construction. Two modules of structural transfer can be used –

1-stage transfer - only one set of transfer rules is used (*.t1x file*).

3-stage transfer - transfer rules are used to group words into chunks, on which later operations such as inter-chunk and post-chunking can be performed (*.t1x, .t2x and .t3x files*).

1-stage transfer can be used for languages which are closely related [6], for e.g. Assamese-Bengali. For Assamese-English we have to use *3-stage transfer* rules. We have used the following transfer files –

apertium-asm-eng.asm-eng.t1x, *apertium-asm-eng.asm-eng.t2x*, *apertium-asm-eng.asm-eng.t3x* for Assamese to English.

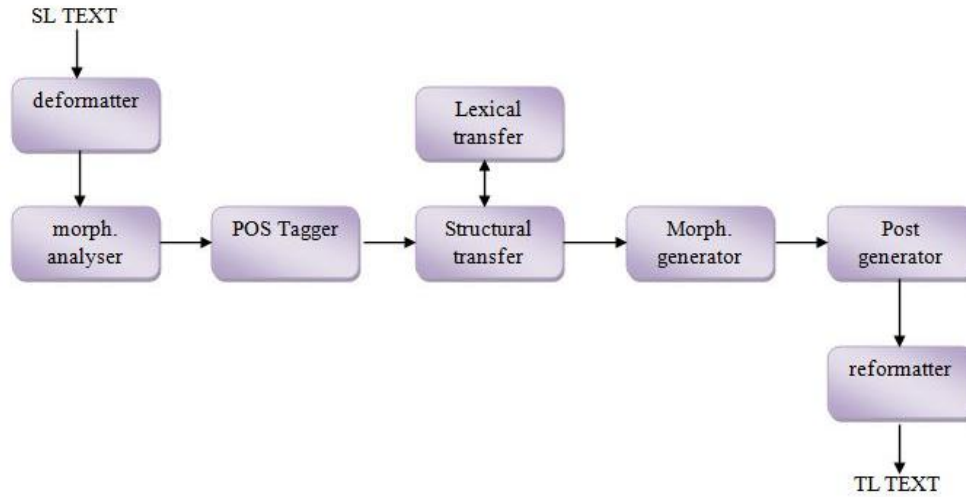
apertium-asm-eng.eng-asm.t1x, *apertium-asm-eng.eng-asm.t2x*, *apertium-asm-eng.eng-asm.t3x* for English to Assamese.

IV. Methodology

A. The Apertium Architecture

The eight module architecture of Apertium [7] is shown in the following figure:

Figure 2: Architecture of Apertium



Deformatter

The deformatter separates text from the format information (html tags, space etc) by encapsulating it with brackets, thus the rest of the module ignores it thinking it as blank space between words [12].

For example the html text দিল্লী ভাৰতৰ ৰাজধানী। would be transformed by the deformatter as দিল্লী []

ভাৰতৰ [] ৰাজধানী।

The deformatter allows only text to be translated from the format information not the html tags [14].

Morphological Analyser

Here text in the surface forms is tokenized and one or more LFs (lexical forms) are delivered from each SF (Surface form). For e.g.

^দিল্লী/দিল্লী<np><top><sg><nom>\$ ^ভাৰতৰ/ভাৰত<np><top><sg><gen>\$

^ৰাজধানী/ৰাজধানী<n><m><sg><nom>\$^।<sent>\$

Here ‘দিল্লী’ is analyzed into lemma দিল্লী, proper noun, toponym, singular and nominative; ‘ভাৰতৰ’ is analyzed into lemma ভাৰত, proper noun, toponym, singular and genitive. The characters “^” and “\$” delimit the analyses for each SF; LFs for each SF are separated by “/”; angle brackets “< . . >” are used to delimit grammatical symbols. The string after the “^” and before the first “/” is the SF as it appears in the source input text [13].

Parts of Speech Tagger

This step chooses one lexical form for those words having more than one.

The output of the tagger is something like this:

^দিল্লী<np><top><sg><nom>\$ ^ভাৰত<np><top><sg><gen>\$ ^ৰাজধানী<n><m><sg><nom>\$^।<sent>\$

Structural Transfer and Lexical Transfer

As the name says ‘Lexical’ changes words while ‘Structural’ changes the word order.

The output of the lexical transfer is:

^দিল্লী<np><top><sg><nom>/Delhi<np><top><sg><nom>\$

^ভাৰত<np><top><sg><gen>/India<np><top><sg><gen>\$

^ৰাজধানী<n><m><sg><nom>/capital<n><sg><nom>\$^।<sent>/.<sent>\$

Structural Transfer is performed in 3 stages: chunking, interchunking and postchunking.

The final output of the Structural transfer stage would be as follows:

^Delhi<np><top><sg>\$ ^be<vbser><pres><p3><sg>\$ ^capital<n><sg>\$ ^of<pr>\$

^India<np><top><sg>\$^.<sent>\$

Morphological Generator

Generate the surface forms of the target language words from the decomposition in lemma plus attributes obtained from the previous steps.

Post Generator

Reformatter

For e.g. “Delhi [*<* is capital of [*>*] India.” is restored into “Delhi *<*is capital of*>* India.”

Apertium basically works on dictionaries and shallow transfer rules. There are 3 basic dictionaries: two monolingual dictionaries and a bilingual dictionary.

1) *The Monolingual Dictionaries*

<?xml version="1.0" encoding="UTF-8"?>

<sdefs>

<sdef n="sg"/>

</sdefs>

I

```
<section id="main" type="standard">
```

</dictionary>

<alphabet>অআইঈঊঋঌঐঔকখগঘঙচছজঝঞটঠডঢণতথদধনপফবভমযবল্লরশষসহস্ফড়়য়ংেঁং</alphabet>

Next we need to add some symbols inside `<sdefs>` tag. The symbols can be noun, pronoun, proper noun, singular, plural, present continuous etc.

<sdef n="n"/>

<sdef n="sg"/>

<sdef n="pl"/>

</sdefs>

<pardefs>

<e><p><l/><r><s n="n"/><s n="nt"/><s n="sg"/><s n="nom"/><r></p></e>

</pardef>

Here the singular form the noun is “অপরাধ” and the plural form is “অপরাধবোৰ”. Here the noun “অপরাধ” is also neuter gender and nominative case.

<section id="main" type="standard">

</section>

In the same way we have created the English monolingual dictionary.

The bilingual dictionary describes the mapping between words [8]. Lexical transfer is performed by the bilingual dictionary i.e. it gives translations between words and not tags. The skeleton of the bilingual dictionary is shown below:

```
<dictionary>
  <alphabet/>
  <sdefs>
    <sdef n="n"/>
    <sdef n="sg"/>
    <sdef n="pl"/>
  </sdefs>
  <section id="main" type="standard">
```

```
</section>
</dictionary>
```

The skeleton is very much similar to that of the monolingual dictionary. Here the paradigm section is not required. We just need add an entry to translate between the two words in Assamese and English, inside the <section> tag. This can be done as following:

```
<section id="main" type="standard">
<e><p><l>অপৰাধ<s n="n"/><s n="nt"/></l><r>crime<s n="n"/></r></p></e>
</section>
```

All the nouns, pronouns, verbs etc. are added to the bilingual dictionary inside the <section> tag likewise.

C. Transfer Rules

Making rules is the toughest part in RBMT based systems. A sound knowledge in the relevant language is very much required in making the grammatical rules. The lexical transfer only gives translations between words. The reordering of words is not performed by lexical transfer. Now we will perform structural transfer, which can change the order of words and can change or add tags on a per-category (group of lemmas) basis. As we are dealing with Assamese and English languages, we have to perform the 3-stage transfer [10]. The stages are:

- **Transfer stage or chunk stage:** Here the words are translated using the bilingual dictionary and grouped and put into chunks (in the .t1x file). The tags in the words can also be added here and removed or made into 'pointers' that points to the tags in the enclosing chunk.
- **Interchunk stage:** In this stage the chunks are reordered, combined and split and chunk tags are changed (in the .t2x file).
- **Postchunk stage:** This is the final stage of structural transfer. Here the chunks are restored (in the .t3x file).

V. Results and Evaluation

A. Results and Statistics

Now as we have the dictionaries and the transfer rules, we can test our system with some sentences. Some translations are not perfect, because of the problem in the transfer rules. We are currently working on adding more and more rules into our system.

The current status of our work is shown in the following table:

Table II Status of our language pair

	Number Entries
Monolingual Dictionary (asm)	3983 words
Bilingual Dictionary (asm-eng)	4341 words
Monolingual Dictionary (eng)	5940 words
Transfer rules (asm→eng)	13 rules
Transfer rules (eng→asm)	11 rules

Some test results for Assamese to English and vice versa are shown in the following tables:

Table III Assamese to English Translation

Translating	Output	Correct Output
জয়পুৰ ৰাজস্থানৰ এখন বিখ্যাত চহৰ।	Jaipur is a famous city of Rajasthan.	Jaipur is a famous city of Rajasthan.
তাজমহল আগ্ৰাত অৱস্থিত।	Taj Mahal is situated Agra.	Taj Mahal is situated in Agra.
জামা মহম্মদ শ্বাহজাহানে নিৰ্মাণ কৰিছিল।	Jama Masjid was built by Shahjahan.	Jama Masjid was built by Shahjahan.
দিল্লী ভাৰতৰ ৰাজধানী।	Delhi is capital of India.	Delhi is the capital of India.
গঙ্গা নদীৰ পাৰত অৱস্থিত উজ্জয়িনী ভাৰতৰ সপ্ততীৰ্থৰ ভিতৰত অন্যতম।	on bank of Ganges river situated Ujjain India of #seven sacred cities of inside #one of the.	Situated near the banks of the river Ganga, Ujjain is one of the seven sacred cities of India.
মমতাজ মহল শ্বাহজাহানৰ পত্নী আছিল।	Mumtaz Mahal is wife of Shahjahan #was.	Mumtaz Mahal was the wife of Shahjahan.
অন্ধ্ৰপ্ৰদেশ ভাৰতৰ এখন ৰাজ্য।	Andhra Pradesh is a state of India.	Andhra Pradesh is a state of India.

Table IV English to Assamese Translation

Translating	Output	Correct Output
Delhi is the capital of India.	দিল্লী #খন ভাৰতৰ ৰাজধানী।	দিল্লী ভাৰতৰ ৰাজধানী।

Assam is a beautiful place.	অসম এক সুন্দৰ #জেগা	অসম এখন ধুনীয়া ঠাই
Kanak Vrindavan is a popular picnic spot in Jaipur.	কনক বৃন্দাবন এক জনপ্ৰিয় #বনভোজ #দাগ *in জয়পুৰ	কনক বৃন্দাবন হৈছে জয়পুৰৰ এখন জনপ্ৰিয় বনভোজ স্থান
Hyderabad is situated in Bangalore.	হায়দৰাবাদ অৱস্থিত *in #বাংগালোৰ	হায়দৰাবাদ বাংগালোৰত অৱস্থিত
Udaipur is a famous city of Rajasthan.	উদয়পুৰ ৰাজস্থানৰ এক বিখ্যাত চহৰ	উদয়পুৰ ৰাজস্থানৰ এখন বিখ্যাত চহৰ
Canada is a vast country.	কানাডা এক বিশাল #দেশ	কানাডা এখন বিশাল দেশ

Here ‘*<word>’ means that the word is not in the *analyser* and ‘#<word>’ means that the word is not in the *generator*.

B.Evaluation

Now we want to see how well our language pairs works in practice. We can see this by calculating the WER (word error rate) and PWER (post-independent word error rate). The tool used for evaluating WER and PWER is *apertium-eval-translator* [11]. The results of the evaluation are shown in the following table:

Table V Evaluation Results

Corpus	WER	PWER
Tourism data	53.80%	36.96%

The scores are not amusing enough, which indicates that our system is still not ready for use in general. Work is still going on the improvement and addition of more transfer rules.

VI. Conclusion

We have tried to implement a bidirectional RBMT system between Assamese and English language. The results obtained from our work are encouraging but not satisfactory. We are working on the transfer rules and increasing the coverage of words in our dictionary. In the future we are looking forward to improve the translation quality of our system and make it a full-fledged MT system. The long-term plan of our project is also to make our system hybrid, taking the good qualities of both RBMT and SMT systems and see if it gives a better translation output.

References

- [1] “List of language pairs”, Available: http://wiki.apertium.org/wiki/List_of_language_pairs
- [2] Antony P.J., “Machine Translation and Survey for Indian Languages”, “The Association for Computational Linguistics and Chinese Language Processing” Vol. 18, No. 1, March 2013, pp. 47-78.
- [3] “Trunk”, Available: <http://wiki.apertium.org/wiki/Trunk>.
- [4] “Ltttoolbox”, Available: <http://wiki.apertium.org/wiki/Ltttoolbox>.
- [5] Carme Armentano-Oller, Rafael C. Carrasco, Antonio M. Corbí-Bellot, Mikel L. Forcada, Mireia Ginestí-Rosell, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Gema Ramírez-Sánchez, Felipe Sánchez-Martínez, Miriam A. Scalco (2006) "Open-source Portuguese-Spanish machine translation", in In Lecture Notes in Computer Science 3960 (Computational Processing of the Portuguese Language, Proceedings of the 7th International Workshop on Computational Processing of Written and Spoken Portuguese, PROPOR 2006), May 13-17, 2006, ME - RJ / Itatiaia, Rio de Janeiro, Brazil. ((c) Springer-Verlag 2006), p. 50-59
- [6] Francis M. Tyers, Jacob Nordfalk, “Shallow-transfer rule-based machine translation for Swedish to Danish”, “Proceedings of the First International Workshop on Free/Open-Source Rule-Based Machine Translation”, 2009, pp. 27-33
- [7] Forcada, M. L., Ginestí-Rosell, M., Nordfalk, J., O’Regan, J., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Sánchez-Martínez, F., Ramírez-Sánchez, G., and Tyers, F. M. (2011). Apertium: a free/opensource platform for rule-based machine translation. *Machine Translation*, 25(2):127–144.
- [8] “Apertium New Language Pair HOWTO”, Available: http://wiki.apertium.org/wiki/Apertium_New_Language_Pair_HOWTO
- [9] “Assamese Language”, Available: http://en.wikipedia.org/wiki/Assamese_Language.
- [10] “Chunking: A full example”, Available: http://wiki.apertium.org/wiki/Chunking:_A_full_example.
- [11] “Evaluation”, Available: <http://wiki.apertium.org/wiki/Evaluation>.
- [12] Roche, E., Schabes, Y.: Introduction. In Roche, E., Schabes, Y., eds.: Finite-State Language Processing. MIT Press, Cambridge, Mass. (1997) 1–65.
- [13] Garrido-Alenda, A., Forcada, M.L., Carrasco, R.C.: Incremental construction and maintenance of morphological analysers based on augmented letter transducers. In: Proceedings of TMI 2002 (Theoretical and Methodological Issues in Machine Translation, Keihanna/Kyoto, Japan, March 2002). (2002) 53–62
- [14] “Transfer Rules Examples”, Available: http://wiki.apertium.org/wiki/Transfer_rules_examples

Acknowledgement

The authors are thankful to the *Department of Information Technology, Gauhati University* for providing us the corpus, which helped us in making the dictionaries and building the RBMT system with Apertium.