# Extending Capabilities of English to Marathi Machine Translator

**Devika Pisharoty[1], Priya Sidhaye[2], Hrishikesh Utpat[3], Sayali Wandkar[4], Rekha Sugandhi[5]**

**[1] Department of Computer Engineering, MIT College of Engineering
Pune, Maharashtra, India**

**[2] Department of Computer Engineering, MIT College of Engineering
Pune, Maharashtra, India**

**[3] Department of Computer Engineering, MIT College of Engineering
Pune, Maharashtra, India**

**[4] Department of Computer Engineering, MIT College of Engineering
Pune, Maharashtra, India**

**[5] Department of Computer Engineering, MIT College of Engineering
Pune, Maharashtra, India**

**Abstract**

Machine Translation is one of the fastest growing research areas in the field of Natural Language Processing, with a special area of focus being Asian languages. Substantial work has been done in the case of Hindi and Bengali. The scope of this paper is to discuss the future scope of machine translation, with specific focus on translation of Marathi – a language spoken by over 70 million people [1]. The process of machine translation can be expanded to include the use of spelling and grammatical checks, intermediate language, sentiment analysis, proverbs and phrases.

***Keywords**: Artificial Intelligence, Natural Language Processing, Generation, Parsing, Machine Translation (MT), Sense Tagging, POS Tagging, WordNet, Interlingua, Word Sense Disambiguation (WSD), Idioms and Phrases, Hindi, Marathi*

## 1. Introduction

The main objective of Machine Translation (MT) is to break the language barrier in a multilingual nation like India. Majority of the Indian population is not familiar with English while most of the information available on web or electronic information is in English. So, to reach out to the common man across various sections, an automatic language translator is important.

There are several approaches to machine translation (MT) that have been developed over the years. They are broadly categorized into direct machine translation, transfer rules based machine translation and Interlingua based machine translation. While several advances have been made in machine translation systems, an optimal solution still remains elusive. MT systems have to deal with inherent ambiguity in natural languages. The linguistic diversity between source and target languages is one of the greatest challenges in machine translation.

This is particularly true of the Indian subcontinent, with India alone having 22 officially recognized languages. These languages can be grouped into four broad categories – the Indo-Aryan languages (E.g.: Hindi, Marathi, Punjabi), the Dravidian languages (E.g.: Tamil, Tulu), the Sino-Tibetan languages (E.g.: Bodo) and the Austra-Asiatic languages (E.g.: Santali) [1].

One such language, where little research has been conducted, is Marathi. Marathi – native to the Western part of India and particularly the state of Maharashtra – is spoken by around 70 million people. It is a derivative of

Sanskrit and is written in the Devnagri script. It constitutes 52 alphabets, 14 vowels and 36 consonants [1].

Consider a case where the content to be translated is given in the form of a document. This document is divided into paragraphs, and these paragraphs into sentences. A generic procedure for Machine Translation (MT) can be defined as: tokenization, lemmatization, parsing, syntax validation, semantic validation, translation, transformation and reconstruction of sentence. This process is discussed in detail below.

The following sections discuss a basic translation mechanism; various features that can be added to extended the capabilities of said mechanism, and describe the operation of the mechanism when these additional functionalities are integrated with it.

## 2. Architectural overview of a basic translator

A basic translation mechanism can be implemented using the following steps [2]:

1. Tokenize the input sentence.
2. Parse the input sentence using a parser based on the source language grammar rules.
3. Tag the sentence according to parts-of–speech (POS).
4. Perform Lemmatization to obtain the roots of words.
5. Perform Word Sense Disambiguation (WSD) on the lemma to understand the exact meaning of the lemma.
6. Use a bilingual dictionary to obtain appropriate translation of the lemmas.
7. Obtain the proper form of the lemma by using inflections.
8. Reconstruct the sentence using a parser based on the target language grammar rules.

However, this process has a large set of inaccuracies. The overall performance and accuracy of the performance can be further announced by including the following extra functionalities:

1. Spell checking.
2. Analysis and translation of Idioms and Phrases.
3. Sentiment Analysis.
4. Use of intermediate language.

In the following sections, these functionalities are discussed.

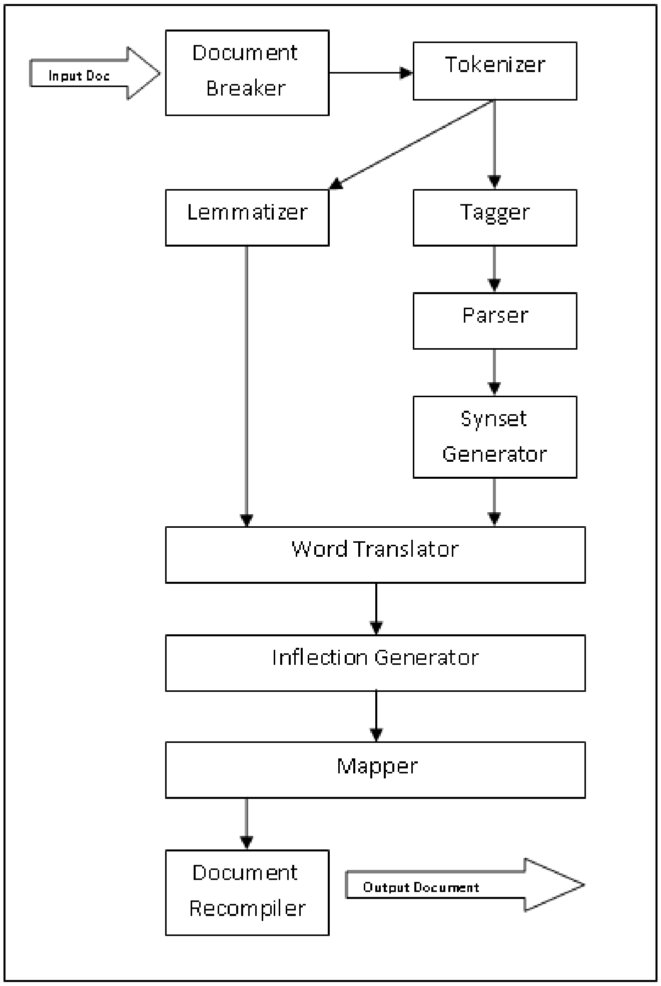


Fig. 1: Representation of a basic translation mechanism

## 3. Spell check

Spell checking is a function that can be executed during pre-processing of the source language sentence. Performing spell checking will improve the operation of the translation, since incorrect tokens will be eliminated from the source language sentence before the inception of the translation process. As a result, the chances of the translation mechanism getting stuck over a misspelt token or incorrect sentence construction will be significantly lowered.

Spelling errors are general of 2 types:

1. Non-word Errors.
2. Real-word Errors.

Non-word Errors imply such errors that result in word constructions that have no meaning. Real-word Errors, on the other hand, result in words that have some meaning in the real word; although they are syntactically incorrect [3]. Both types of errors result mainly from typographical mistakes.

E.g.: The word “boolk” instead of “book” is a non-word error. The word “he” in the sentence “The book lay on *he* table” is a real-word error, with a typographical error causing the word “he” to be substituted in place of “the”.

Most modern spell checkers tend to use word lists. Word lists are basically sets of words belonging to that language [4]. By comparing each token in the source language text with the word list, it can be ascertained if the token belongs to the language, or is a spelling mistake.

It is relatively easy to build such word lists. By scanning large volumes of texts in that particular language (for instance, literature books), the required corpus can be built. However, it is very difficult to expect any word list to be completely exhaustive – mainly due to the fact that languages are continuously evolving. One way of overcoming this shortcoming is by periodically scanning contemporary texts sets such as newspapers or conversations between individuals and adding new words that occur frequently to the word list. The word list searching process can be further optimized by arranging the word list alphabetically and by using hashing techniques [3].

The use of a word list can be used to address non-word errors. However, addressing real-word errors is more complicated. Techniques such as N-gram analysis can be used to detect spelling mistakes [3]:

The detected spelling error, inter alia, can be brought to the user’s notice by highlighting the concerned token. A list of recommended spelling corrections can also be supplied along with.

The process of error correction makes use of a concept known as edit distance. Edit distance is defined as the number of steps taken to edit an incorrect word to its correct form [3].

E.g.: For transforming blook to book, the number of editing steps is 1 i.e. elimination of ‘l’.

Since the number of typographical errors per word rarely exceeds 2 or 3, the list of words within editing distance greater than or equal to 3 shall be displayed along with the misspelt word.

Thus, the inclusion of spellchecker in the translation mechanism will remove typographical errors prior to the processing of the source language text, and thereby improve the performance of the translation system.

## 4. Idioms and Phrases

An integral part of any language is proverbs and sayings, which are sometimes unique to the language. They evolve in accordance with the natural development of any language, and reflect the cultural paradigms integral to that language. These idioms, taken at their face value do not make linguistic sense. As a result, translating idioms and phrases is a complex process. Quite often exactly corresponding phrases in the target language do not exist, thus further complicating the translation process.

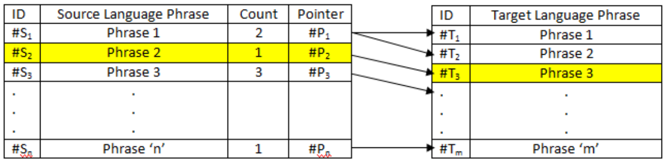
E.g.: Consider the Marathi phrase, नाचता येईना अंगण वाकडे. (“Naachta yeina angan wakde.”) which means, “Despite being unable to dance, blames it on the courtyard saying that it is crooked”. Translating this to English, the phrase roughly corresponds to “A bad workman blames his tools”.

The process of translation of phrases cannot be done in complete un-supervision. At least partial supervision of the system is necessary. To overcome this issue completely, an exhaustive list of proverbs in the source language, equivalent proverbs in the target language, and if an equivalent proverb does not exist, a suitable translation of the meaning of the words must be built. For proverbs that exist in English, but do not have a proverb with an equivalent meaning in Marathi, a statement that conveys the appropriate meaning must be substituted [5].

E.g.: For the proverb "Fools rush in where angels fear to tread", there is no well-known proverb that can be substituted. Thus the meaning of the statement must be put instead of a literal translation.

Handling idioms and phrases will require a separate database and tables to be able to accurately translate proverbs and sayings that appear in the source language. However, thought must be given to the number of occurrences of such sayings in the text. If the text is not rich with proverbs, a huge database will be rendered unnecessary. In such a case, some proverbs that are statistically known to occur more in text can be included in the database.

A proposed translation mechanism for phrases is as follows:



| ID | Source Language Phrase | Count | Pointer |
|---|---|---|---|
| #$S_1$ | Phrase 1 | 2 | #$P_1$ |
| #$S_2$ | Phrase 2 | 1 | #$P_2$ |
| #$S_3$ | Phrase 3 | 3 | #$P_3$ |
| .<br>.<br>. | .<br>.<br>. | | |
| #$S_n$ | Phrase ‘n’ | 1 | #$P_n$ |

| ID | Target Language Phrase |
|---|---|
| #$T_1$ | Phrase 1 |
| #$T_2$ | Phrase 2 |
| #$T_3$ | Phrase 3 |
| .<br>.<br>. | .<br>.<br>. |
| #$T_m$ | Phrase ‘m’ |

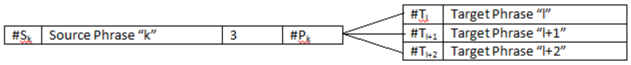
Fig. 2: Representation of translation of phrases

The translation mechanism consists of 2 tables consisting of the source language phrases and the target language phrases respectively. The table *“source_language”* consists of the source language phrase, the number (count) of the corresponding phrases in the target language and a pointer to the entry of the first phrase in the target language. The table *“target_language”* consists of the corresponding target language phrases.

The count column is included in *“source_language”* since a single phrase in the source language may map to more than one phrase in the target language. The entry in the pointer column is used to access the first corresponding phrase from *“target_language”* and successive entries equal in number to *(count)* are retrieved.

E.g.: The English phrase, “From the frying pan into the fire” translates to – “आधीच उल्हास त्यातून फाल्गुन मास” ("Aadhich ulhas tyatun falgun maas") and “आधीच दुष्काळ त्यातून ठणठण गोपाळ” ("Aadhich dushkaaL tyatun thaNthaN gopaL").

On translation of each sentence of the source language, the tokens are linearly analysed to check if it maps onto a known phrase. Once a known phrase is encountered, its entry in *“source_language”* is accessed and the corresponding set of target language phrases from *“target_lemma”* is obtained. The most accurate phrase from the set of target phrases can be obtained using the actual words in the source phrase. WSD techniques can be used here.



| | | | |
|---|---|---|---|
| #$S_k$ | Source Phrase “k” | 3 | #$P_k$ |

| | |
|---|---|
| #$T_l$ | Target Phrase “l” |
| #$T_{l+1}$ | Target Phrase “l+1” |
| #$T_{l+2}$ | Target Phrase “l+2” |

Fig. 3: Representation of translation of a single phrase

E.g.: Consider the phrase “As you sow, so shall you reap” existing in the table *“source_language”* with index number “$S_i$”. Entry $S_i$ in *“source_language”* maps to “पेरावे तसे उगवते” ("Perawe tase ugawate"). The target language phrase is substituted into the target language text.

To further speed up this process, phrases that occur more frequently in the source language can be placed at the beginning of the table *“source_language”*, thus decreasing the number of rows that would have to be searched to arrive at a match. Hashing can also be implemented to speed up the search process.

Thus, translation of phrases and idioms will be done in parallel with the translation of the source language text and thereby improve the operational accuracy of the translation mechanism.

## 5. Sentiment Analysis

Sentiment analysis is the process of mining opinions, positive, negative, or neutral. Since a lot of information is in the form of opinions about something, this type of analysis is useful to classify documents. It can be put to use in the process of translation as well.

Sentiment analysis is basically is refereed to using 3 parameters [6]:

1. Determination of subjectivity - Whether a text is objective or subjective.
2. Polarity – Whether it is positive or negative.
3. Strength – Whether it is strongly or weakly positive/negative

Based on the degree of supervision in the system, two sentiment analysis methods are proposed: a word-based or semantic approach, or a machine learning (ML) approach. The word-based approach uses dictionaries of words tagged with their *semantic orientation* (SO), and calculates sentiment by aggregating the values of those present in a text or sentence. The ML approach uses collections of texts that are known to express a favorable or unfavorable opinion as training data, and learns to recognize sentiment based on those examples [6]:

The sentiment to be conveyed from a sentence generally revolves around a few keywords in that sentence. Studies have shown that sentiment can be communicated using a set of frequently occurring words. However, these set of words are to a large degree domain-specific and also depend on the original author of the source language text.

E.g.: The phrases “That is one awesome bike!” conveys that a strongly positive sentiment regarding the object i.e. the bike via the term “awesome”. However, here the domain of the sentence has to be considered to ascertain the sentiment being expressed.

Sentiment analysis is essentially done as a positive/negative analysis. So some key words are taken into account while doing this. Words like good, amazing, nice show positive, words like awful, standard, mundane give a negative turn. Apart from this, they also use words which give a flow to a document, like thus, though, however show a change in emotion within the document. These words can be used to obtain the overall sentiment of the document.

Instead of simply classifying sentiment as being "positive" or "negative", it can be further classified using more complex means. Three algorithms - Naive Bayes classification, maximum entropy classification, and support vector machines – have shown varying degrees of efficiency.

Thus, by using sentiment analysis, a deeper understanding of the sentiment of the text can be obtained. This would be beneficial while performing word sense disambiguation, and also obtaining a broader understanding of the document as a whole. This added information can be used wherever needed.

## 6. Use of Intermediate Language

An intermediate language is particularly useful when implementing multi-lingual translation. By using an intermediate language for representing the contents of the source language text and translating from the intermediate language to the target language, a much wider range of parallel corpora are more efficiently used [8]. A higher degree of independence between the source language and target language is also achieved by decoupling their interdependence.

Theoretically, given a set of 'm' source languages that can be translated into an intermediate language "X", and a set of 'n' target languages that the language "X" can be translated into, a single translator can be built support (m.n) languages.

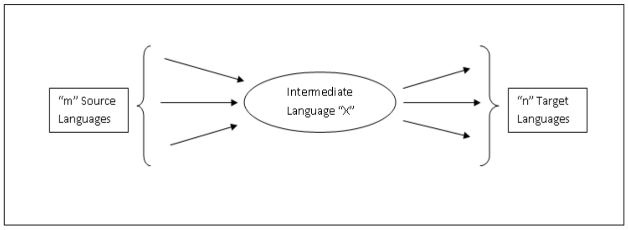


Fig. 4: Use of Intermediate Language

This is particularly effective when focusing on Indian languages. All Indian languages are classified into 4 families based on their language of origin [1]. Of these, Sanskrit is the oldest and most syntactically rich language. If a language translators being built is considered such that it has source language as English and target languages as Hindi and Marathi, using Sanskrit as an intermediary language will be very useful. To be able to combine translators and extend them to various other Indian languages, Sanskrit is an ideal choice. This interlingua approach can also be extended on the source language side. Many European languages have originated from Latin. Thus, Latin can be used as a second intermediate language and the translator will be able to include many European and Indian languages.

However, one constraint that has to be considered here is loss of information during translation. The information contained in an English sentence is very less compared to the information contained in a Sanskrit sentence. In order to bridge this information gap, certain assumptions will have to be made, and these assumptions will result in the introduction of inaccuracies. The translation of Sanskrit to the target language is a relatively easier task, since the Sanskrit sentence (i.e. intermediate sentence) is rich in information.

Thus, the use of intermediate languages will enhance the scope of the translation mechanism.

## 7. Summary of Advanced Translation Mechanism

As per the above discussion, the basic translation mechanism can be modified to include greater functionality as follows:

1. Break the sentence into individual sentences.
2. Tokenize the input sentence.
3. Check each token for spelling mistakes.
4. Pass the tokens to the idioms and phrase translator.

5. Parse the input sentence using a parser based on the source language grammar rules.
6. Tag the sentence according to parts-of–speech (POS).
7. Lemmatized to obtain the roots of words.
8. Perform sentiment analysis on the sentence.
9. Perform Word Sense Disambiguation on the lemma to understand the exact meaning of the lemma.
10. Represent the tokens in the intermediate language.
11. Use a bilingual dictionary to obtain appropriate translation of the lemmas.
12. Obtain the proper form of the lemma by using inflictions.
13. Reconstruct the sentence using a parser based on the target language grammar rules.

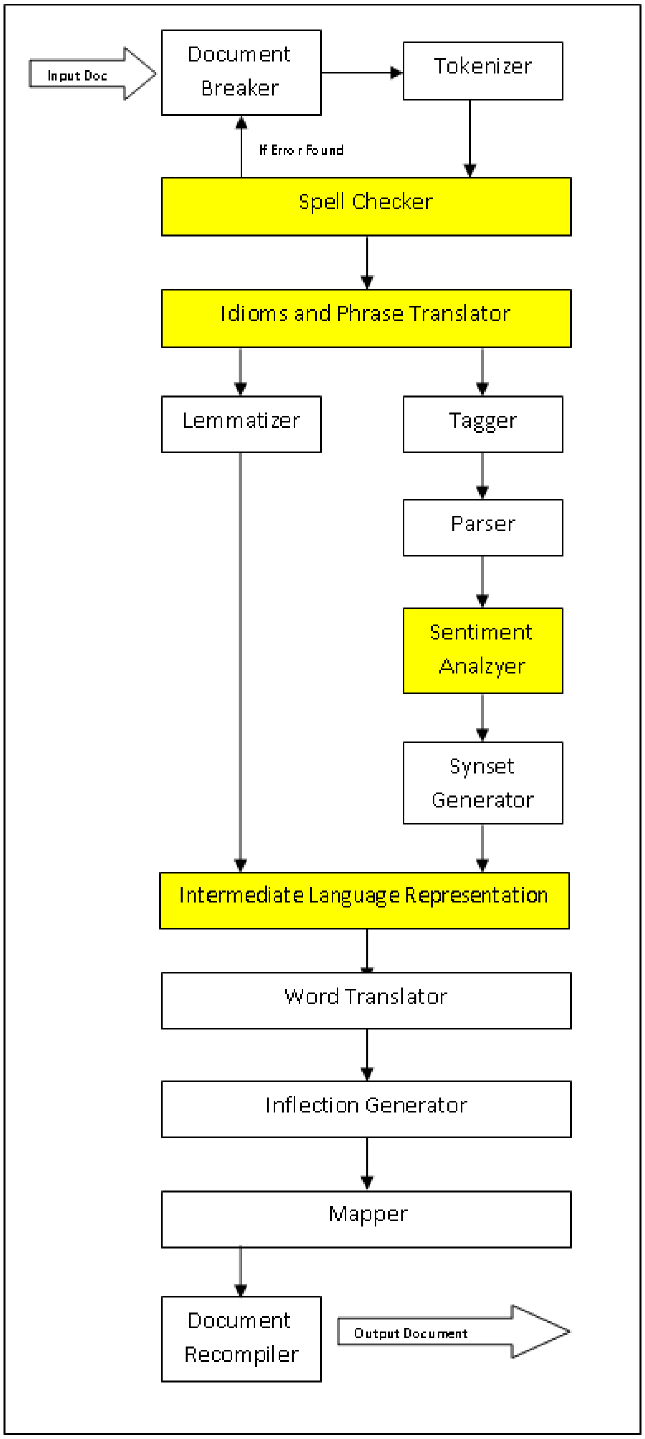


Fig 5: Representation of translation mechanism with added functionality

## 8. Conclusions

Thus, by adding the above mentioned functionalities, the performance of the translation mechanism can be significantly improved. Although this process may increase the time requirement of translation, it will increase the accuracy of the translation. Coupling this with the increased processing capabilities of today's machines, the above mentioned additions would be a trade-off that would have a significant improvement of the basic translation mechanism.

### Acknowledgments

The authors would like to thank MIT-College of Engineering -Pune and Board of College and University Development (BCUD), University Of Pune, for the grant for conducting the research work for multi-lingual machine translation system, under which this paper has been written. This paper is a result of the ongoing work for the said research topic.

## References

[1] Ljiljana Dolamic, "*Influence of Language Morphological Complexity on Information Retrieval"*, La Faculte des Sciences de l'Universite de Neuchatel, 2010.

[2] Rekha Sugandhi, Devika Pisharoty, Priya Sidhaye, Hrishikesh Utpat, Sayali Wandkar and, Rajendra Khope, *Managing Tokens for Machine Translation from English to Marathi*, International Journal of Engineering, Science and Research (IJESR), Volume 1, Issue 3, October 2011

[3] Mansour Sarr, "Improving Precision and Recall Using a Spellchecker in a Search Engine", Department of Numerical Analysis KTH and Computer Science, Stockholm Royal Institute of Technology, Stockholms Universitet.

[4] Harold L. Somers, "*Machine Translation and Minority Languages"*, Department of Language Engineering, UMIST.

[5] Martin Volk, "*The Automatic Translation of Idioms – Machine Translation vs. Translation Memory System".*

[6] Julian Brooke, Milan Tofiloski, Maite Taboada, *"Cross-Linguistic Sentiment Analysis: From English to Spanish",* Department of Linguistics, Simon Fraser University, Burnaby, BC, Canada.

[7] Bo Pang, Lillian Lee, Shivakumar Vaithyanathan, "Thumbs up? Sentiment Classification using Machine Learning Techniques", Department of Computer Science, Cornell University, USA, IBM Almaden Research Center, San Jose, USA.

[8] Trevor Cohn and Mirella Lapata, *"Machine Translation by Triangulation: Making Effective Use of Multi-Parallel Corpora"*, Human Computer Research Centre, School of Informatics, University of Edinburgh.