

# भारती – An Attempt to Cross Lingual Search

Asha Bharambe<sup>1</sup> Hitesh Narang<sup>2</sup> Naveen Pahuja<sup>3</sup> SnehaJaisinghani<sup>4</sup> SundeepNawani<sup>5</sup>

<sup>1,2,3,4,5</sup>Department of Information Technology

<sup>1,2,3,4,5</sup>Vivekanand Institute of Technology, Mumbai

**Abstract**— In this paper we describe भारत<sup>ती</sup>, a cross-lingual search application which would be used to crawl the data on the web in the assumed Indian National language viz. Hindi. This work is done with the intent of helping people to get access to the world of internet who doesn't know English language. भारत<sup>ती</sup> will accept the input in Hindi language, determines the part of speech tags of each word from the sentence, sends the input to Google Translate Server which translates the input in to English language except all nouns which if translated will change the meaning of search query using its Translate API , as Google Translate API is one of the best translator among all other available translators, then it searches for the translated text by using Google search Engine and crawls the data on the web using JSOUP technology, the crawled results will be in English and will be translated back in Hindi by the Google Translate Server using its Translate API and will be displayed accordingly.

**Keywords:** API, Google Translate Server, BHARATI

## I. INTRODUCTION

Internet has revolutionized our lives. However, most of the information on the internet being in English causes the internet to be effectively unavailable to the rural masses unqualified in English. The benefits of IT have not been derived by a large section of Indian population, mainly in rural areas. The reasons are given as lack of infrastructure, inadequate dissemination of information and so on. However, the problem of language barrier should be cited as one of the primary reasons.

The need for multilingual information processing is enormous for a country like India. Most of the advanced information for the every domain should be in local Indian languages. This should be available on the web for the rural people to read, assimilate and use.

भारती presents a meaningful search technique that would search the pages written in English language and present the output to the user in the Indian language i.e. Hindi. Therefore, भारत<sup>ती</sup> will be very useful for the people in rural India whose basic means of communication is Hindi language. With this application billions of bytes of information on web will be easily available and understood by them and thereby they can take the benefits of the web.

The modules involved in the भारत<sup>ती</sup> are Translation of input, POS Tagging, Searching, Crawling the data on the web, Translation of searched results, displaying the results. The Translation of input and Searched results will be done using Google Translate API, POS Tagging will be done by using Argmax computation and Bayes Theorem from theory of Probability, Searching will be done through the Google search engine, Crawling will be done by using JSOUP technology, and Java Technology will be used for taking input and displaying the output.

## II. PROBLEM STATEMENT

The existing search engines like Google, Yahoo, Bing, etc. are giving the accurate results in the English language as shown in the figure 1. All these existing search engines give inappropriate results in Hindi language and all other Indian languages. Therefore, to avoid this particular drawback, we first translate the input text in English language, search the results in English, translate the results in Hindi and then display the results accordingly. If we search the following text shown in Example A on existing search engines like Google, Bing, etc. in English language, we get more accurate results. While if we search the same particular text in Hindi language on the same existing search engines, we get less accurate results as shown in figure 2. Let us understand this by considering Example A and B. Example A - "Cinemas in Mumbai"

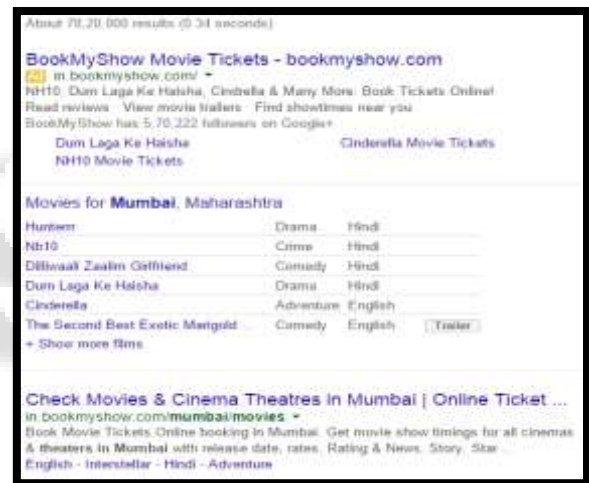


Fig. 1: Results Obtained in English

Example B – "मुम्बईकेसिनेमाघर"



Fig. 2: Results Obtained in Hindi

From these results, we observe that we are getting more accurate results in English language as compared to the results found in Hindi.

#### A. Objectives

The objective of the project is to use the generic platform components and be able to provide a meaningful search which will allow:

- The search to be executed in Indian Language.
- Presenting the accurate search results in the said Language.

#### B. Existing Systems

We have done a small literature survey on the existing Search Engines, Translators and after the survey; we found some of the features as very useful and interesting. At the same time, these existing Systems also have some drawbacks.

One of the most common feature which is found in all the existing search engines like Google, Yahoo, Bing, etc. is that all these existing search engines gives most accurate results in the English language as compared to all the other Indian languages. The common drawback found in all these existing search engines is that they give irrelevant and inappropriate search results in Indian languages like Hindi, Marathi, etc. whereas if we search the same for the same results in English, most accurate results are found.

Similarly, we have done small literature survey on the existing translators and found that most of the times the translator gives relevant translation results. One of the drawback found in the existing translators is that they are not able to identify the correct parts of speech tags for each word in the sentence. Let us consider one example which shows that existing Google Translator is not able to identify the common nouns; if we give input to Google Translator as “आशाएकअच्छीशिक्षिकाहै”, the output obtained in English is “Hope is a good teacher” which shows that Google Translator is not able to identify proper nouns correctly as shown in figure 3 below.



Fig. 3: Google Translator Not Able to Identify Proper Nouns.

As existing search engines always give the most accurate search results in the English language, therefore we will translate the input (which is in Hindi) to English by using the Google Translate API as it is one of the best available translator.

As Google Translator is not able to identify the proper nouns correctly, therefore we include POS Tagging Algorithm to avoid this particular drawback.

So Before Translation, the input will be given to the POS Tagging Algorithm which will find the correct parts of speech tags for each word of the sentence and then the entire input sentence will be given to translator to translate the sentence in English so that we should get accurate search results in Hindi.

### III. METHODOLOGY

#### A. Translation and Searching

The figure 4 shows the high level architecture diagram of भारती which explains that at client side there will be user interface and small patch code for POS Tagging will also be encapsulated inside the application. At client side, after accepting the input query and finding the part of speech tags of each word in the sentence using POS Tagging algorithm (except all nouns which have exactly same meaning in English), input query will be sent to the server side i.e. to the Google Translate Server for translating the input in English.

Then, भारती will search and crawl the results in English itself and finally the searched results will be translated back in input language i.e. Hindi and displayed accordingly. The search results for the query may include some websites designed in Indian language; they will remain same as no translation will be required because the page itself is written in Indian language. Modules executed at the client side are accepting input in Hindi, deciding POS tags for each word in sentence, displaying the search results in Hindi. Modules which are executed at server side i.e. at Google server are Translating input using Google Translate API, searching for the results through Google search engine. This gives the overview of implementation of accepting the input, Translation of input, Searching, Displaying the results. Next, in this paper, we will discuss the implementation of the algorithm used for the POS Tagging.

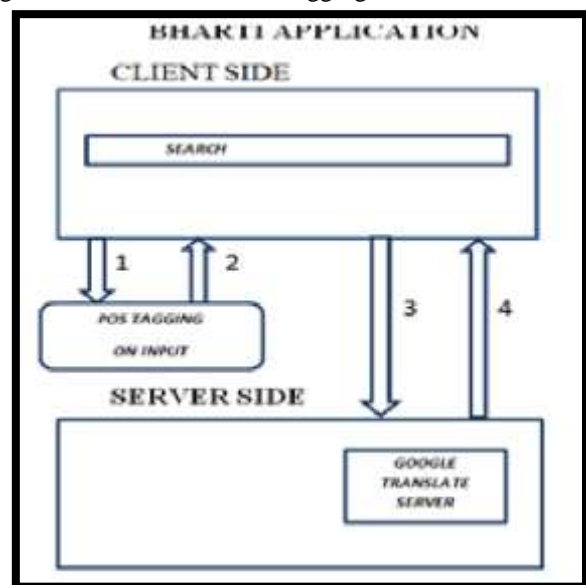


Fig. 4: High Level Architecture of भारती

### B. POS Tagging Algorithm

Part of Speech Tagging is a process that attaches each word in a sentence with a suitable tag from the given set of tags. The set of tags is called as tag set. The Standard tag set for English Penn Treebank.

Process to carry out POS Tagging:

- List all possible tags for each word in a sentence.
- Choose best suitable tag sequence.

To design an algorithm for POS tagging, the first important step is to list all possible tags for each word in a sentence. Process to perform this particular step is as follows;

**भारती** will be using the Hindi corpora (large textual data) which will act as a training set for POS tagging process.

Consider the sentence “<sup>^</sup>प्रगती ट्रेन के निकलने का समय. राजधानी के आने का समय.” as a corpus to understand the working of algorithm.

The tags which we are considering for this corpus are:

N = Noun, V = Verb, P = Preposition, <sup>^</sup> = start of sentence, . = end of sentence

But, actually for Hindi language there are 24 tags.

Let say for example; end user searches for query: “प्रगती ट्रेन के आने का समय”

The output obtained from tagged corpora is:

प्रगती can be noun or verb, ट्रेन can be noun, के can be preposition, आने can be verb, का can be preposition, समय can be noun.

Total number of tags = 3

If a sentence contains only a single word in it then every tag sequence will have only one tag in it.

Different possible tag sequences:

<sup>^</sup>NNPVPN. , <sup>^</sup>VNPVPN.

Now, we have to choose the best possible tag sequence among them. The process to choose the best possible tag sequence will be described in detail, but before that we will understand the role of argmax computation and Bayes Theorem in POS Tagging algorithm.

In POS Tagging Algorithm, we will make use of Argmax computation which is given as follows:

$T = \text{Argmax}(P(t_i | w_i))$  where T represents best possible tag sequence.

$T = \text{Argmax}(P(t_i) * P(w_i | t_i)) \rightarrow$  (By Bayes Theorem) [ignoring  $P(w_i)$ ]

Bayes theorem acts as a filter for bad tag sequences and helps us to take advantage of prior probability.

Where  $P(t) = P(t_i | t_{i-1}) =$  Bigram Probability;  $t_i =$  current tag and  $t_{i-1} =$  previous tag.

In Bigram Probability, Probability of the current tag depends completely on the probability of its previous tag.

And

$P(w_i / t_i) =$  Lexical Probability.

Here we find the probability of a word given any appropriate tag at that position.

For every possible tag, the bigram probability and lexical probability is computed and product of both the probability values is computed.

Finally, as per the rule of Argmax computation, that particular tag will be considered whose product of bigram and lexical probabilities will be the highest. In this way, the best possible tag sequence will be decided by using Argmax computation and Bayes theorem.

Now, Let us understand the process of deciding the best possible tag sequence.

Now, we have different possible tag sequences, among them we have to obtain the best possible tag sequence. For this we will consider the first tag from all the possible tag sequences and we will obtain the Bigram probability and Lexical probability values and the product of them. Finally, that value of product will be considered whose value will be highest. Similarly, we will obtain the best possible tag for the second word and so on.

Considering the first possible tag sequence obtained as output from the training corpora; i.e. <sup>^</sup>NNPVPN.

First tag of first tag sequence is N;

Its Bigram Probability =  $P(N | \text{^}) =$  Probability of N tag given its previous tag is <sup>^</sup> =  $\text{count}(N \text{ tag is preceded by } \text{^}) / \text{count}(\text{All } \text{^} \text{ tags in corpus}) = \#(\text{^}, N) / \#(\text{^}) = 2 / 2 = 1.$

Its Lexical Probability =  $P(\text{प्रगती} | N) =$  Probability of word at that position is प्रगती given that tag at that position is N =  $\text{count}(\text{प्रगती being N in corpus}) / \text{count}(\text{All N in corpus}) = \#(\text{प्रगती}, N) / \#(N) = 1/4.$

Product of bigram and lexical probability =  $P(N | \text{^})$

\*  $P(\text{प्रगती} | N) = 1 * 1/4 = 1/4.$

Now let us consider the first tag of the second possible tag sequence i.e. V.

Its Bigram Probability =  $P(V | \text{^}) = 1/2.$

Its Lexical Probability =  $P(\text{प्रगती} | V) = 1/3.$

Product of bigram and lexical probability =  $P(V | \text{^}) * P(\text{प्रगती} | V) = 1/2 * 1/3 = 1/6.$

As the product obtained when considering the first tag as noun is greater than product obtained when considering first tag as verb, the best possible tag for the first word of the sentence is N (noun). All the remaining tags for the remaining words of the sentence in both the possible tag sequences are same. So the best possible tag sequence for the sentence is “NNPVPN”. In this way, best possible tag sequence is decided among all the available tag sequences.

While implementing this, we will be using the Graph Data Structure. The edges of graph will contain the product of bigram and lexical probability values and the edge which will have the highest weight i.e. the highest value of product, that edge will be considered for further process. In this way, process of deciding the best possible tag sequence has now been reduced to the Graph Traversal Algorithm as shown in figure 5.

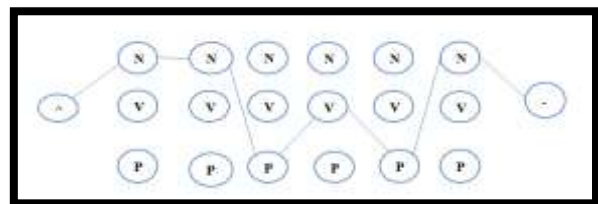


Fig. 5: Graph Traversal Showing Best Possible Sequence

1) *Algorithm*

- (1) Find out all the possible tags for every word of the sentence from the corpus.
- (2) Find out Bigram probability of each tag from each tag sequence obtained from corpus.
- (3) Find out Lexical probability of each tag from each tag sequence obtained from corpus.
- (4) Find out product of bigram and lexical probability values.
- (5) Consider that tag whose product of bigram and lexical probability values is highest.
- (6) Use graph traversal algorithm to decide the best possible tag sequence.

IV. CONCLUSION

In this paper we have discussed the importance of the Indian National Language viz. Hindi in the rural areas as they don't understand the English. It is observed that 5 – 6% of Indian population do not know the English language. Therefore **भारती** gives benefits of IT to large section of Indian population, mainly in rural areas. It will be very useful for the people in rural India whose basic means of communication is Hindi language.

In conclusion we found a solution for Hindi search application with the good accuracy rate as compared with other search engines by using;

- POS Tagging
- Google Translate API
- Crawling using JSOUP

REFERENCE

- [1] <http://www.mkyong.com/java/jsoup-send-search-query-to-google/>
- [2] <https://www.youtube.com/watch?v=aeOLjFe256E&list=PLD392E2ACAEF0C689>
- [3] <http://textofvideo.nptel.iitm.ac.in/106101007/lec1.pdf>
- [4] <https://developers.google.com/console/help/new/>
- [5] <https://translate.google.co.in/>
- [6] <http://textofvideo.nptel.iitm.ac.in/106101007/lec10.pdf>
- [7] <http://textofvideo.nptel.iitm.ac.in/106101007/lec11.pdf>
- [8] <http://textofvideo.nptel.iitm.ac.in/106101007/lec12.pdf>