ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

Evaluation of Similarity metrics for translation retrieval in the Hindi-English Translation Memory

Shashi Pal Singh[1], Ajai Kumar[2], Hemant Darbari[3], Sukanya Chauhan[4], Nandita Srivastava[5], Priya Singh[6]

AAI, Center for development of Advanced Computing, Pune, India[1,2,3]

Banasthali Vidyapith, Rajasthan, India[4,5,6]

Abstract: This paper discusses the bigram model with 3-operation Edit Distance (Levenshtein Distance) String matching metrics for translation retrieval in Hindi-English Translation memory system. In this method we used the statistical language modeling (N-gram approach) to compute bigrams and then implemented the dynamic programming algorithm Levenshtein Distance to find the minimum number of edit operations required transforming one bigram to another and will act as a measure to provide extent for the matching of current input and the source in the TM. This measure will decide whether the translation retrieved correspondingly will be exact match or fuzzy match. Other string matching approaches are evaluated with Levenshtein Distance proving to be more effective comparatively.

Keywords: String Matching, Bigram modelling, Levenshtein Distance, Hindi-English Translation memory (TM).

1. INTRODUCTION

A Translation memory (TM) [10] [12] technology is used as a translator's aid for providing a good precision translation. Basically, a translation memory is a database that consists of Translation unit (TU) which constitutes the language pair of source and its translation. TM system matches the source language input to be translated with the source in the translation unit and retrieve the translation candidates in target language which may help human translator to either accept the translation, replace it with a fresh translation or modify the translation to fit into the new translation and update it in TM.TM system uses String similarity metrics[3] so as to retrieve the target language corresponding to the source language in TU by computing source language similarity with the current input.TM system should provide fast access to the TM and at the same time retrieve the similar translation candidates with great accuracy[11]. Therefore, the access and retrieval speed and accuracy should be evaluated using different string similarity metrics.

This paper focuses on the string similarity metrics employed so as to have proper translation retrieval around the Hindi to English TM[7][8] and this is achieved by employing the N-gram modelling[16] approach, string similarity function and threshold. The bigram model [13] is used to consider the local context or the character order or maintain co-occurrence of words in the Hindi sentences and the obtained bigram phrase of the current input sentence are further matched with the bigrams of the source sentence in TM. Similarity function, used to match two phrases, is Levenshtein distance [1] using matrix which is the character based operation. This metric reflects the lexicographical matches. The more the correspondence, the smaller the distance and the one with the exact match has a score 0. The available exact or fuzzy matched sentences will be retrieved as the translations to the translator. In exact matching, the current input sentence is completely matched with the source sentence in the TM while in the fuzzy matching the sentence is not completely matched and retrieves the phrases that are matched with the differences marked.

2. MATCHING TECHNIQUES

In past years, lot of research work has been done to syntactically compare two strings. String similarity measure states the extent how much the two stated strings are similar or dissimilar. These algorithms can be effectively used in the fuzzy string searches that will constitute the main approach for the best match selection in Hindi-English TM if not an exact match is found.

In this section we briefly discuss the widely used metrics such as the Cosine similarity, Dice coefficient, Jaccard Coefficient and Edit Distance with different edit operations.

2.1 Cosine Similarity

It is a vector based method to measure the similarity between the string pair S1 & S2. The strings are transformed in vector space and each term is represented as an axis. [2] Cosine of the angle between vectors S1 and S2 states the similarity. If the value comes to be 0 then angle between the vectors is 90 and if the value is 1 then the vectors are equal with different magnitude.

$$\text{Cos}(\overrightarrow{S1}, \overrightarrow{S2}) = \frac{\overrightarrow{S1} \cdot \overrightarrow{S2}}{|\overrightarrow{S1}||\overrightarrow{S2}|} \quad [2]$$

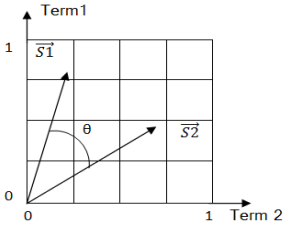

Figure 1 Cosine Similarity evaluation Sim (S1, S2) =cos (θ)

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

This approach does not care about the word ordering in string.

2.2 Dice Coefficient

It is a word based similarity measure and is named after Lee Raymond Dice. For strings S1 & S2, it is defined as:-

$$DC = \frac{2|S1 \cap S2|}{|S1|+|S2|}$$ [3]

For example:-

S1="राम खाना खाता है"

S2="सीता खाना खाती है"

$$DC = \frac{2*(2)}{4+4} = 0.5$$

2.3 Jaccard Coefficient

Jaccard coefficient also measures the similarity based on word without retaining their order i.e. it is a token based measure and is defined as the ratio of the common words to the owned by both strings.

$$JC = \frac{|S1 \cap S2|}{|S1 \cup S2|}$$ [1]

For the above strings:-

JC=2/6=0.33

2.4 Edit Distance

It is measure of the minimum number of edit operations required to transform one string S1 into another S2.It uses a dynamic programming paradigm[3].It is a character based similarity measure and includes the following edit operations:-

- Insertion
- Deletion
- Replacement
- Transposition

Depending on which edit operations are used, Edit Distance can be classified as:-

2.4.1 Levenshtein Distance

It is defined as the minimum number of edit operations i.e. Insertion, Deletion & Replacement to transform one string into another. This method is best suited for phrases or sentences of small length. Figure 2 shows the matrix to compute the Levenshtein Distance between "प्रजातंत्र" and "लोकतंत्र" which results to be 5.

2.4.2 Hamming Distance

It includes the edit operation of replacement only to transform one string into another.The length of the string remains constant after transformation.It is used for error detection and correction in the strings.

2.4.3 Damerau–Levenshtein Distance

It includes the edit operations of Insertion, Deletion, Replacement and Transposition to transform one string into another.

		ल	ो	क	त	ं	त	ृ	र
	0	1	2	3	4	5	6	7	8
प	1	1	2	3	4	5	6	7	8
ृ	2	2	2	3	4	5	6	7	8
र	3	3	3	3	4	5	6	7	8
ज	4	4	4	4	4	5	6	7	8
ा	5	5	5	5	5	5	6	7	8
त	6	6	6	6	5	6	5	6	7
ं	7	7	7	7	6	5	6	6	7
त	8	8	8	8	7	6	5	6	7
ृ	9	9	9	9	8	7	6	5	6
र	10	10	10	10	9	8	7	6	5

Figure 2 Levenshtein Distance computation

3. PROPOSED METHODOLOGY

Having the Hindi-English Translation Memory(TM) and the current Hindi input to be translated, we first have to segment and then compute the bigrams of the current input and the source of the TM using N-gram approach to model order of the characters in the similarity. Further the extent of the common bigrams between a given string pair determines how much similar they are.

Definition 1:- The Similarity function used employs dynamic programming paradigm, Levenshtein distance, which constructs a matrix m for given strings S1 and S2, having |S1|+1 row and |S2|+1 columns and fill the entries correspondingly in the matrix cell.
Initially, m [i, 0] =i for i=1 to $|S_1|$ and m [0, j] =j for j=1 to $|S_2|$.

For i=1 to |S1| and m [0, j] =j for j=1 to |S2|, the m[i, j] (Levenshtein distance) computed is [1]: -

m[i, j]=min{ m[i, j-1]+1,
m[i-1, j]+1,
m[i-1, j-1]+c}, where c=0 if
S1[i]=S2[j] else c=1.

Algorithm 1 shows the pseudo code of Levenshtein Distance algorithm.

Algorithm 1: [1] Levenshtein Distance
Input- two strings S1 and S2
Output- score of similarity

1. int m[i, j]=0
2. for i← 1 to |S1|
3. do m[i, 0]=i
4. for j← 1 to |S2|
5. do m[0, j]=j
6. for i ← 1 to |S1|
7. do for j← 1 to |S2|
8. do m[i, j]= min{m[i-1, j-1]+if(S1[i]=S2[j]) then 0 else 1,
   m[i-1, j]+1,m[i, j-1]+1}
9. Return m[|S1|,|S2|]

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

Time complexity in computation is O(|S1|x|S2|) and the space complexity is O(min(|S1|,|S2|)).

Definition 2:- N-grams [6] are the substrings of size N. A window of size N is slide over the sentences to partition into substrings of length N.
In our approach, we used the bi-grams, which are the substrings of size 2. The sentence is partitioned into substrings of length 2. Sentences in the TM are partitioned into bigrams and then correspondingly matched with the bigrams of the query using string similarity metrics.

Formally our TM retrieval uses score ranking measure:

$$\text{Score} = \max\left(\frac{1\text{-edit dist}(S_1, S_2)}{|S_1\text{bigrams}|}, 0\right),$$

The following example illustrates the matching approach used in our TM System:
Let S1 is the string of current input that needs to be translated and S2, S3 are the strings of the source in TM.

For the string S1="भारत भौगोलिक दृष्टि से विश्व में सातवाँ सबसे बड़ा देश है"

The bigram sequences returned are {"भारत भौगोलिक", " भौगोलिक दृष्टि", " दृष्टि से", " से विश्व", " विश्व में", " में सातवाँ", " सातवाँ सबसे", " सबसे बड़ा", " बड़ा देश", " देश है"}

For the string S2=भारत जनसंख्या की दृष्टि से विश्व में " "सबसे बड़ा देश है दूसरा

The bigram sequences returned are {"भारत जनसंख्या", "जनसंख्या की", "की दृष्टि", "दृष्टि से", "से विश्व", "विश्व में", "में दूसरा", "दूसरा सबसे", "सबसे बड़ा", "बड़ा देश", "देश है"}

For the string S3= भारत भौगोलिक दृष्टि में विश्व सबसे" "सातवाँ से बड़ा है देश

The bigram sequences returned are {"भारत भौगोलिक", "भौगोलिक दृष्टि", "दृष्टि में", "में विश्व", "विश्व सबसे", "सबसे सातवाँ", "सातवाँ से", "से बड़ा", "बड़ा है", "है देश"}

The number of bigrams matched between the strings corresponds the similarity between the two.

Matching of S1 and S2 is demonstrated in Figure 3.

In our employed similarity metrics, the sentence S1 if matched corresponding to S2 results out to be 60% while corresponding to S3 it results to be 20% matched. Since the sentence S3 is not ordered grammatically and has no semantic interpretation thus scoring low, the sentence S2 will be considered as an appropriate translation and will be retrieved.

Translations corresponding to the source in TM having matching score within the specified threshold will be retrieved as a suggestive list to the translator. The string pairs having greater matching score is given priority in comparison to other pairs.

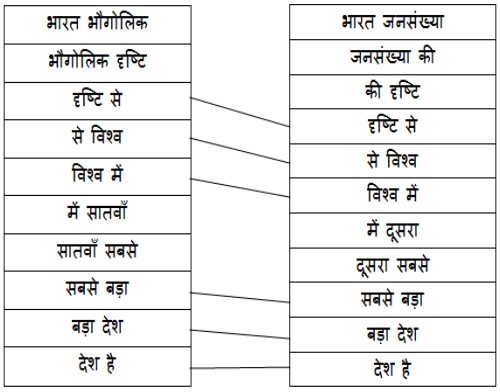

Figure 3 Match of 6 bigrams out of 10, states 60% matching

Matching of S1 and S3 is demonstrated in Figure 4.

Figure 4 Match of 2 bigrams out of 10, states 20% matching

4. EVALUATION AND EXPERIMENTAL STUDY

We have developed a bilingual corpus of 10,000 sentences as a Hindi-English Translation memory. We randomly obtain the 100 sentence pairs as a test case. In order to obtain the accuracy of our approach, we evaluated its performance and compare it with other existing techniques.

4.1 Evaluation of our technique

There are many Hindi words that have minor phonetic differences like "वे and वो", "हम and हमें", "है and हैं" etc, that corresponds to same sense and have the same translation in English. Similarly for gender cases like "रहा and रही", "खाता and खाती" that are treated differently in Hindi but have the same translation in English . These words will be treated differently in the token based or word based approaches and will be considered dissimilar while using Edit Distance approach they will have very less percentage of mismatch.

For example:-
The Hindi sentences in 1 have the same English translation so for query of any either of them the same translation should be retrieved.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

Table 1: Hindi Sentences with their English Translation

S.No.	*Hindi Sentences*	*English Translation*
1	वे लोग आएँगे| वो लोग आएँगे|	Those people will come.
2	मैं अकेला हूँ| मैं अकेले हूँ|	I am alone.
3	आप ठीक हो? आप ठीक हैं?	Are you alright?
4	आप कैसे हो? आप कैसी हैं?	How are you?
5	मैं जा रहा हूँ| मैं जा रही हूँ|	I am going.

These sentences in 1 will be approximately matched using edit distance approach as it will just require the substitution of "े" by "ो" while using the token based approaches the words "वे" and "वो" will be treated as different tokens and regarded as mismatch. Similarly, for the sentences 2 & 3, edit distance approach will act as an effective similarity metrics in comparison to the token based approaches and will provide approximately exact matching. In sentences 4 & 5, the gender disparity between Hindi and English is effectively handled using edit distance similarity metrics as in Hindi some words like "रहा" & "रही" refers to the same English translation and just require the substitution of "ा" by "ी" instead of treating them as different tokens.

4.2 Comparison with existing technique

We evaluated various other string matching approaches in our research and the obtained results concluded that our used approach gives more precise results comparatively.

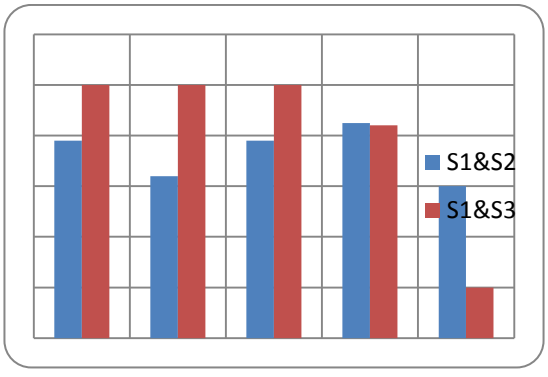

Figure 5 Performance evaluation of our used approach

5. CONCLUSION

In this paper we have discussed the string similarity metrics employed in our Hindi-English Translation Memory for translation retrieval and studied how it outstand other similarity metrics in our purpose. We extended the Levenshtein Distance technique by employing the bigram modeling which maintains the local context or word order in the sentences. Experimental studies state that our used approach is highly efficient and productive for our purpose and performs better than other discussed techniques.

REFERENCES

[1] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. 2008. Dictionaries and tolerant retrieval in An Introduction to Information Retrieval, Cambridge University Press.
[2] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. 2008. Scoring, term weighting and the vector space model in An Introduction to Information Retrieval, Cambridge University Press.
[3] Hao Chen. 2012. String Metrics and Word Similarity applied to Information Retrieval, Master Thesis, School of Computing, University of Eastern Finland.
[4] Sachin Anklekar and Chetan Shah. 2006. Anusmriti: A Language Independent Translation Memory, Technical Report CM-KBCS-2006-1, C-DAC, Mumbai.
[5] Steve Iverson, "Working with Translation Memory", Multilingual Computing & Technology, Volume 14, Issue 7.
[6] William B. Cavnar and John M. Trenkle. 1994. N-Gram-Based Text Categorization, Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval.
[7] N. Joshi, I. Mathur, S. Mathur. 2011. Translation Memory for Indian Languages: An Aid for Human Translators, Proceedings of 2nd International Conference and Workshop in Emerging Trends in Technology.
[8] Nisheeth Joshi & Iti Mathur. 2011. Design of English-Hindi Translation Memory for Efficient Translation, Proceedings of National Conference on Recent Advances in Computer Engineering.
[9] Felix Naumann.2013. Similarity measures, Hasso plattner Institute.
[10] A.Zerfass. 2002. Evaluating Translation Memory Systems. Proceedings of the LREC 2002 Workshop, Las Palmas, Canary Islands, SPAIN.
[11] Timothy Baldwin & Hozumi Tanaka. 2001. Balancing up Efficiency and Accuracy in Translation Retrieval. Journal of Natural Language Processing vol. 8.
[12] Soroosh Mortezapoor. 2013. Translation Memory and Bitext Alignment. Seminar in Artificial Intelligence.
[13] Graham Neubig. Bigram Language Models. Nara Institute of Science and Technology (NAIST).
[14] Fei Song & W. Bruce Croft. 1999. A General Language Model for Information Retrieval. Proceedings of the 22nd annual international ACM SIGIR Conference on Research and Development in Information Retrieval.
[15] Lynn E. Webb. 1999. Advantages And Disadvantages Of Translation Memory: A Cost/Benefit Analysis. Monterey Institute of International Studies Monterey, California.
[16] D. Jurafsky, J. H. Martin, Speech and Language Processing. 2008. An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Pearson Education.

BIOGRAPHIES

Mr. Shashi Pal Singh is working as STO, AAI Group, C-DAC, Pune. He has completed his B.Tech and M.Tech in Computer Science & Engineering and has published various national & international papers. He is specialised in Natural Language Processing (NLP), Machine assisted Translation (MT), Cloud Computing and Mobile Computing.

Mr. Ajai Kumar is working as Associate Director and Head, AAI Group, C-DAC, Pune. He is handling various projects in the area of Natural Language Processing,

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

Information Extraction and Retrieval, Intelligent Language Teaching/Tutoring, Speech Technology [Synthesis & Recognition ASR], Mobile Computing, Decision Support Systems & Simulations and has published various national & international papers.

Dr. Hemant Darbari is working as Executive Director in C-DAC, Pune. He is one of the founding members of C-DAC, an R&D institute set up by the Department of Electronics and Information Technology; Govt. of India for carrying out advanced research in new and emerging technological domains. He has to his credit, 85 Technical Papers that have been published in national & international Journals & Conference Proceedings.

Nandita Srivastava received the B.Tech degree in Computer Science and Engineering from H.N.B Garhwal University, Uttarakhand in 2013 and recently pursuing M.Tech in Computer Science and Engineering from Banasthali University, Rajasthan. Her research interests are in Information Retrieval, Natural Language Processing, Artificial Intelligence and Pattern Recognition and Image Processing.

Priya Singh received the B.Tech degree in Information Technology from S.I.E.T (U.P Technical University), Uttar Pradesh in 2013 and recently pursuing M.Tech in Computer Science and Engineering from Banasthali University, Rajasthan. Her research interests are in Natural Language Processing, Artificial Intelligence and Geographic Information System.

Sukanya Chauhan received the B.Tech degree in Computer Science and Engineering from P.C.S.T (Rajiv Gandhi Proudyogiki Vishwavidyalaya), Madhya Pradesh in 2011 and recently pursuing M.Tech in Computer Science and Engineering from Banasthali University, Rajasthan. Her research interests are in Natural Language Processing, Artificial Intelligence, Geographic Information System and Soft Computing.

DOI 10.17148/IJARCCE.2015.4813