CAMES, **29**(1–2): 125–144, 2022, doi: 10.24423/cames.395

This article belongs to the *Special Issue on Advanced Optimization Methods for Uncertainties in Intelligent Industrial Systems* edited by Dr. I.J. Jacob, Dr. B.H. bin Ahmad and Dr. Z.F. Khan

A Method to Integrate Word Sense Disambiguation and Translation Memory for English to Hindi Machine Translation System

Sunita RAWAT

Department of Computer Science and Engineering
Shri Ramdeobaba College of Engineering and Management
Nagpur, India; e-mail: ssunitarawatt@gmail.com

Word sense disambiguation deals with deciding the word's precise meaning in a certain specific context. One of the major problems in natural language processing is lexical-semantic ambiguity, where a word has more than one meaning. Disambiguating the sense of polysemous words is the most important task in machine translation. This research work aims to design and implement English to Hindi machine translation. The design methodology addresses improving the speed and accuracy of the machine translation process. The algorithm and modules designed in this research work have been deployed on the Hadoop infrastructure, and test cases are designed to check the feasibility and reliability of this process. The research work presented describes the methodologies to reduce data transmission by adding a translation memory component to the framework. The speed of execution is increased by replacing the modules in the machine translation process with lightweight modules, which reduces infrastructure and execution time.

Keywords: machine translation, word sense disambiguation, statistical machine translation, translation memory.

1. Introduction

The task of translating by a computer from a given human language to another language is called machine translation (MT). Actually, a computer is unable to process natural languages correctly. As languages are continuously evolving, polysemous, and irrational, it is very difficult for the machine to handle all such qualities. That is the reason why processing natural languages and their MTs are very difficult. From the initial days of the computer, the importance of automatic disambiguation of word senses for the information available has been a concern. Almost all natural languages have ambiguous words with more than one meaning. Human languages are inherently ambiguous, and it is a generic phenomenon to have more than one sense for a word. In natural language processing, the most essential task is to disambiguate the word sense. This is also required in many fields such as information retrieval, lexicography, information extraction, question answering, semantic interpretation and so many but most appropriately in MT. Because of this, it is necessary to apply the technique of word sense disambiguation (WSD) so that ambiguity of the word gets resolved [7, 23, 37].

WSD approaches can be classified as supervised techniques and unsupervised techniques. The problem with supervised WSD techniques is that they need huge data for training the system and generating good results [34–36]. Creating a proper database that contains English words and their meaning in Hindi is a very expensive, time-consuming and tough job. As language is continuously evolving, the process of adding new domains, writing words involved in that domain and their corresponding meanings must get repeated.

However, in the unsupervised WSD technique, there is no need to maintain such a database. The unsupervised WSD technique uses the knowledge to advise the meaning of the word instead of using the corpus [6, 29]. So, whenever there is a need to perform WSD without data learning the unsupervised WSD technique is appropriate.

The most apparent application of WSD is MT. The reason for considering WSD for MT is that if sense identification of the words in the given context is correct for the source language, then the prediction of the accurate meaning is more in the target language. Statistical machine translation (SMT) is used for MT in that the first training is conducted and based on that the system learns and translates the given input statements. However, the problem with this MT technique is that it needs vast data for training and requires more processing time. Therefore, such a method is not suitable for time-critical applications and corpus of small size. SMT cannot analyze input text syntactically or semantically. To find the sequence of words, which gives the best translation, SMT applies statistics obtained throughout model training.

In this article, publicly available tourism domain sense-tagged parallel corpus is used [28]. To find the translation of the input sentence, the SMT system uses statistical models to find the sentence with maximum probability. The MTS module responsible for performing this task is known as a decoder. Alignment plays a crucial role while performing translation, as it undergoes preprocessing.

The arrangement of corpus sentences is referred to as sentence-level alignment. In the presented research, an English-Hindi parallel corpus is used. Here, the sentences are said to be aligned when the sequence of a particular sentence in the English file is present in the same sequence in the Hindi file translation of that sentence. There is a requirement of a properly aligned parallel corpus to train the SMT system. In [16, 32], the authors show the word-level alignment by the Cartesian product of the words' positions. It can be picturized by writing a sentence pair (one sentence in English and another in Hindi) and placing lines linking the words. In SMT, the training of the statistical models is done by only word-level alignment. So, it is a very important part of the system.

The paper is organized as follows. Section 1 covers the introduction of word sense disambiguation and SMT. Section 2 discusses the work done by other researchers. Section 3 introduces different MT techniques. Section 4 discusses the methodology used and system architecture. Section 5 presents implementation details of the translation memory. Experimental results are discussed in Sec. 6. Section 7 presents the comparative analysis and last is the conclusion.

2. Related work

The exact meaning of an ambiguous term is chosen by considering the given context. A survey of a number of methods related to this is presented in [1]. There is so much research on word sense disambiguation using the supervised technique (see, for example, [14, 18, 20, 27, 31]). Even though there are several choices of learning methods, it is observed that classifiers such as naïve Bayes produce good results compared to other methods. This classifier is very competitive and works well on the feature selection process. In some literature (for example, [12, 21]), the parallel corpus was used to solve the WSD problem. All these methods focused on finding the most accurate sense in source linguistic content; in this research, the main concern is to find the correct sense in the given query and translate it to the required language. As ambiguity is resolved on the source side then its translation will be correct on the target side.

There is a lot of work conducted on WSD and it can be used in real-time applications such as information retrieval, question answering, mono-lingual and multilingual search engines, etc. However, the most appropriate application of WSD is MT. Various studies on WSD found that applications based on MT provide good results [3, 21]. Applying WSD to a monolingual information is considered difficult, while applying it to a bilingual content turns out to be even more difficult. In this scenario, the sense of the ambiguous word should be identified correctly so that it is translated correctly.

It is a known fact that sense annotated corpora, parallel corpora and wordnet are rare data resources. As the most appropriate application for WSD is multilingual MT where these resources are needed. In [3], the authors applied a WSD approach that can work on specified language even in the absence of its sense tagged corpora. The concept behind this was that languages rich in resources would help languages that do not have enough resources in the same domain. For example, one language can project its parameters in corpus and wordnet to another language that lacks them.

In [5], the authors explored whether the accuracy of MT can be improved by using WSD. The authors applied the WSD technique to Chinese-English language pairs with the aim to find the correct translation of Chinese text into English. It was assumed that ambiguities of words should be resolved according to the context in which they are mentioned. By calculating pairwise similarity, the nearest neighbors of the input text from the data on which it trained were recognized. The best translation into English was identified by taking the majority "votes" of the nearest neighbors.

Word alignment is a fundamental requirement for SMT [17, 24, 26]. Automatic fetching of multilingual text is very easy after using word-aligned corpora [2]. In the translation of language, ambiguities in word meaning are disseminated in another way. Hence, WSD is one of the areas of research in SMT [9]. In [11], the authors discussed grammar induction and different methods of utilizing statistical word alignment for it. If quality of word alignment is good, then several applications of natural language processing (NLP) generate better results. Researchers have worked on several automatic word alignment methods. In [15], the authors use one-to-one mapping of words, i.e., one word from the source language will get map with one word from the target language; this concept is covered under the competitive linking algorithm (CLA). Some techniques such as IBM Models 1–5, HMM [25] and LEAF [10] are based on a probabilistic model. Still, in many studies on data alignment researchers use IBM and HMM models. GIZA++ is a tool built by using above mentioned techniques.

Since MT came into the picture, researchers have developed and worked on many MT techniques. The objective is to make contents of one language available to other. Translations may be available in the form of a word to word or corpus-based. Example-based MT system is a good choice of translation if the text to be translated or very similar text is already available in the translation memory. Example-based MT is not that successful if it will not find close matches in the translation memory. The SMT system works exactly opposite to this. It mostly translates the given input sentence exactly even if the same sentence is not provided in the training [13, 33].

In [4, 30], MT from the English language to the Hindi language is done using the statistical method. The translation system consists of three units. First is the language model used to compute the probability of a given text with respect to the target language. The language model is given as input to the decoder. Second is the translation model used to calculate the probability of the target sentence with respect to the source text. Last, the decoder is used to find the translation text with the highest probability. A parallel dataset of size 5000 is used to train the SMT model. The accuracy of SMT depends on the quality of corpus and learning of parameter estimation. Google translator also makes use of SMT. It has a vast collection of corpus and learns SMT parameters from that corpus. In [8], the authors have worked on parameter estimation and mathematics of SMT.

In an example-based MT system, three steps are needed to translate a given text. In the first step, the availability of the input text is checked in bilingual corpus. In the second step, the useful string is extracted from the matched sentence. In the third step, the translated text is recombined [22].

Statistical models are used for producing the best promising results. Some researchers have recognized that instead of a single MT system a hybrid MT system has more benefits.

3. MT approaches

Table 1 represents various MT approaches with techniques used by each approach. The technique used by each approach is explained in short points by explaining how it works, what are the rules followed by these methods and which approaches can be combined together.

Table 1. MT approaches with their characteristics.

Approach	Technique
Supervised corpus-based approach	In supervised machine learning, system is trained on the annotated corpus. Basic data is used for solving further complexity.
Unsupervised corpus-based approach	Unsupervised methods induce senses by clustering word occurrences or contexts. Infer cross-language senses distinctions by using aligned parallel corpus.
Knowledge-based approach	Hand-coded rules are used for removing ambiguity. To filter out the inconsistent senses, selectional restrictions are used. It compares dictionary with respect to context. For finding similarity score it uses semantic similarity measures. It uses the one-sense-per discourse technique.
Hybrid approach	A knowledge-based can be combined with the parallel corpus (aligned). A knowledge-based approach can be combined with the unsupervised approach. A knowledge-based can also be used in supervised word sense disambiguation for searching purpose.

4. WSD using naïve Bayes technique

The MT field is one of the prime domains of machine learning, with consistent changes in the design level algorithms, tools, technology, deployment and access methods. The new concept of WSD is designed taking into consideration the context of MT process. In this work, the probabilistic concept of naïve Bayes is used for disambiguating the meaning of words in the source language. The probability of words (in a given language) being aligned with their corresponding words (in the target language) is calculated. With the help of Giza++ tool, the bilingual statistical dictionary is created over a parallel corpus (tourism domain). Probabilities are predicted based on such statistical dictionaries. This research also describes an approach of reusing the translated components, thereby reducing the load on the framework used in MT. The results generated are encouraging, as good and positive change is observed compared with the existing systems. In the existing system, there are no concepts of translation memory while conducting MT, so those systems took more time for translation. In those types of systems, if some or similar statement comes for translation, again and again, it will follow the entire process of translation, which takes more time.

Open-source tools such as GIZA++, Moses, etc., are used in design to reduce the overall cost of the software product. The dataset of "tourism" available on the website of Kendriya Vidyalaya, Indian Institute of Technology (KV IIT) in Mumbai is used for training and testing the system. This dataset is used for training and testing the system.

Figure 1 shows the system architecture. First of all, input query (in English) is preprocessed. This query is then searched in the translation memory (called translation lookup); if the query is found in the translation memory, its translated segment is shown as an output. Otherwise, this query is translated by the SMT. After translation, source and target segments are stored in the translation memory for future use, and the translated segment is shown as the output.

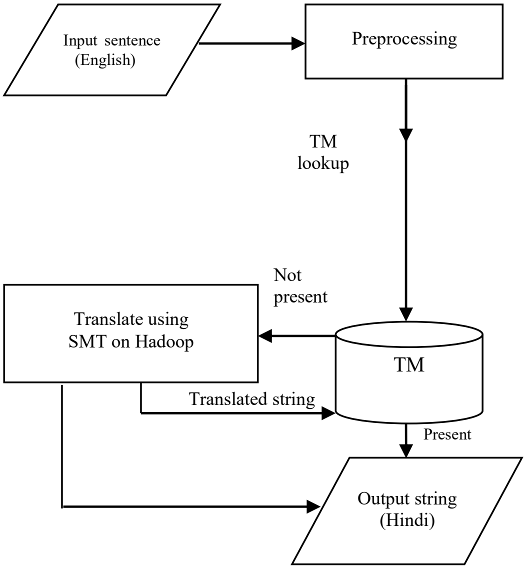


Fig. 1. The system architecture.

Algorithm

```
Input query
  1. Preprocessing of input query.
  2. Find match of input query in TM.
  3. If the same segment is available in TM.
         Then show its corresponding translated segment as output
       Else
         Translate the query using the SMT system and show the translated
         segment as output and at the same time save this input query and its
         translated segment in TM as a pair.
```

4.1. *N*-gram model

This model is helpful to perform training. In a specified sentence, contiguous order of n words is known as n-gram. In an n-gram, if the value of n is 1 it is known as a "unigram"; if the value of n is 2 it is known as a bigram; if the value of n is 3 it is known as a trigram and so on. For instance, consider the sentence: Lata and Anuradha are best singers. If a bigram is applied to the considered sentence, i.e., $N = 2$, the n-grams will be as follows:

- Lata and,
- and Anuradha,
- Anuradha are,
- are best,
- best singers.

In this study, trigrams are used. After identifying the n-gram, morphological analysis is done for the input words. The parallel corpus English-Hindi (Tourism) is used to obtain the consequent Hindi words for the English words. The sequence of words is calculated by the naïve Bayes algorithm. This algorithm provides the series of words with the maximum word order.

4.2. The probabilistic model

Word alignment is the basic requirement for SMT. From the aligned segment pairs, SMT creates its own translation rules. The alignment of words in the aligned sentence is nothing but forming the link between the target words and the source words. There can be more than one alignment for a given English sentence and its translated Hindi sentence. A probability value is assigned to each alignment. Let us consider an alignment s, source sentence e and target sentence h. The probability of s given e and its translation h is expressed in the following form:

$$P(s|e,h) = \frac{P(s,h|e)}{P(h|e)}. \tag{1}$$

Summing over s, on both the sides gives:

$$\sum_s P(s|e,h) = \sum_s \frac{P(s,h|e)}{P(h|s)}. \tag{2}$$

Summation of LHS is equal to 1, and hence the translation model is expressed as:

$$P(h|e) = \sum_s P(s,h|e). \tag{3}$$

Therefore, the probability $P(h|e)$ of the translation model is the summation of all probabilities of generating an output Hindi string h and an alignment s given an input English string e. To create a translation model GIZA++ uses the estimate maximization algorithm on parallel data.

Consider an English sentence denoted by S_E. Suppose $S = \{c_1, c_2, ..., c_k, ..., c_{k+2}, ..., c_{|S|}\}$ is the n-gram representation of SE (whereas c_k is the ambiguous word). Therefore, there will be N number of translations of c_k, $\{d_1^k, d_2^k, ..., d_N^k\}$. Here, the aim is to obtain a suitable translation for the ambiguous term c_k. A proper explanation of the classifier i as follows:

$$p(d_i^k|S) = p(d_i^k|c_1, c_2, ..., c_k, ...), \tag{4}$$

$$p(d_i^k|c_1, c_2, ...c_k, ...) = \frac{p(d_i^k)p(c_1, c_2, ..., c_k, ...|d_i^k)}{p(c_1, c_2, ..., c_k, ...)}. \tag{5}$$

Our goal is to find the argument that maximizes $p(d_i^k|S)$; hence, the denominator calculation can be skipped. Equation (6) shows how Eq. (5) is approximated:

$$p(d_i^k|c_1, c_2, ..., c_k, ...) \approx p(c_1, c_2, ..., c_k, ...|d_i^k). \tag{6}$$

It is required to apply the chain rule for calculating Eq. (6). Let us consider the words present in the given query as not dependent. Therefore, Eq. (6) can be given by Eq. (7):

$$p(d_i^k|c_1, c_2, ..., c_k, ...) \approx \prod_{j=1}^{|S|} p(c_j|d_i^k). \tag{7}$$

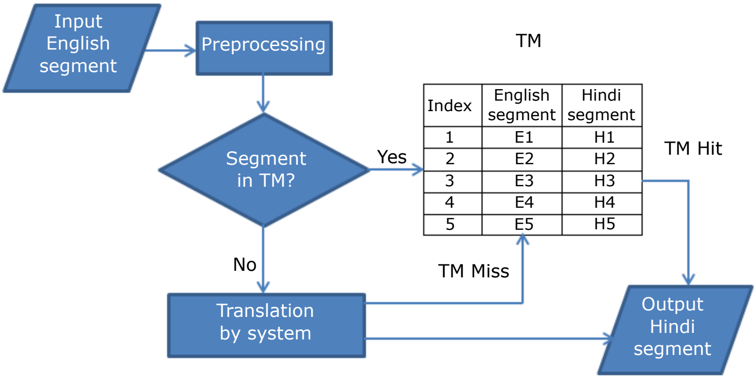
Equation (8) gives the best meaning after calculating the multiplication of all translation probabilities without considering the place of the ambiguous word:

$$\text{BestSense}_w(S) = \arg\max{}_{d_i^k} \prod_{j=1}^{|S|} p(c_j|d_i^k), \tag{8}$$

where $i = 1, ..., N$.

5. Translation memory (TM)

It is essentially a database that stores previously translated content in a pair of source-target manner. In the future, if the same translation is needed, as it is stored in the repository, it can be taken from TM instead of translating it again. So, it saves the efforts of translation as well as time. Figure 2 represents the translation process using TM. Here, TM is a data repository that contains English text with its translated Hindi text in pair. Input query will be an English text that first undergoes preprocessing and then tries to find a match with the already available English text in TM. It may find a 100% match or fuzzy match. Depending on the similarity score, it represents the result. If it is a 100% match (TM Hit) then the corresponding Hindi sentence is shown as the output. If it is a fuzzy match (TM Miss), the translation is required to be done by the SMT system. After translation by the SMT, finally, the pair of English segment and its newly translated Hindi segment is stored into TM, and the Hindi segment is shown as the output.



Index	English segment	Hindi segment
1	E1	H1
2	E2	H2
3	E3	H3
4	E4	H4
5	E5	H5

Fig. 2. Translation process.

Below, we present the difference in working steps of TM Hit and TM Miss:

Steps in TM Hit

1. English segment is given as an input by the user.
2. Preprocessing is done.
3. Availability is checked in TM (available).
4. The corresponding Hindi segment is retrieved and shown as the output.

Steps in TM Miss

1. English segment is given as an input by the user.
2. Preprocessing is done.
3. It is checked in TM (not available).

4. Now the translation is done by SMT.
5. TM is updated with new segment pair.
6. Translated segment (Hindi) is shown as the output.

The use of TM makes the system more efficient as it improves the translation speed by saving the processing time. If the size of TM keeps increasing, chances of getting a sentence in TM are better. For that reason, as the size of TM increases translation speed of the system will also increase. However, there is one problem, as the size of TM keeps on increasing a TM lookup or searching some data will take more time. Therefore, to sort out this problem the concept of indexing is used in this work. Table 2 represents some English-Hindi segments in translation memory.

TABLE 2. Presentation of English – Hindi segments in translation memory.

No.	Sentence in English	Sentence in Hindi
1	Delhi is capital of India	दिल्ली भारत की राजधानी है
2	Dhruv works in a bank	ध्रुव बैंक में काम करता है
3	Jaipur is a tourist place	जयपुर एक पर्यटक स्थल है
4	Jaipur is a pink city	जयपुर एक गुलाबी शहर है
5	Hawa-Mahal is located in Jaipur	हवा महल जयपुर में स्थित है
6	She sat on the river bank	वह नदी के किनारे बैठी थी

$$T_1(Q_1) = [Q_{1i}, Q_{2i}, Q_{3i}, Q_{4i}, ..., Q_{ki}], \tag{9}$$

T_1 is the instance when query Q_1 is given, where $Q_{1i}, Q_{2i}, Q_{3i}, Q_{4i}, ..., Q_{ki}$ are the words present in the query.

This $T_1(Q_1)$ is given to SMT and $T_1^S(Q_1)$ is generated as:

$$T_1^S(Q_1) = [Q_{1i}^S, Q_{2i}^S, Q_{3i}^S, Q_{4i}^S, ..., Q_{ki}^S]. \tag{10}$$

T_1^S is the instance when query Q_1 is translated by SMT, where $Q_{1i}^S, Q_{2i}^S, Q_{3i}^S, Q_{4i}^S, ..., Q_{ki}$ are the SMT translated words of the given query.

The mapping of words between a given query and the translated query is done in two ways:

1. Referring to dictionary;
   To map the Q_i^{ST} it is given to dictionary and can find Q_i^{ST}.
2. Indexing in TM.
   To map the Q_i^T it is searched in TM with the indexing technique and can find Q_i^{ST}.

6. Results

6.1. Experimental results

The system is validated using precision, recall and F-Score. A total of 15 276 sentences from the parallel English-Hindi corpus are considered. As shown in Table 3, six experiments are done by varying the number of sentences considered for training and testing.

Table 3. Experimental data.

Experiment no.	Sentences considered for training	Sentences considered for testing
1	3000	12 276
2	6000	9276
3	9000	6276
4	10 693	4583
5	12 000	3276
6	14 000	1276

In each experiment, 50 queries (Q1, Q2, ..., Q50) are translated and translation is evaluated by calculating precision, recall and F-Score. As shown in Table 4, the English query is represented as E1, E2, ..., E50, and the Hindi query is represented as H1, H2, ..., H50. If exact query is found in TM, then

Table 4. Results of experiment 1.

Query	English query	Hindi query	Source	Time [sec]	Precision	Recall	F-Score
Q1	E1	H1	Server	0.111	0.500	0.425	0.459
Q2	E2	H2	TM	0.068	0.525	0.346	0.417
Q3	E3	H3	TM	0.046	0.200	0.125	0.154
Q4	E4	H4	Server	0.251	0.636	0.653	0.644
Q5	E5	H5	TM	0.039	0.050	0.042	0.046
Q6	E6	H6	Server	0.061	0.523	0.452	0.485
Q7	E7	H7	Server	0.305	0.423	0.322	0.366
Q8	E8	H8	TM	0.104	0.038	0.028	0.032
Q9	E9	H9	TM	0.069	0.067	0.034	0.045
Q10	E10	H10	TM	0.057	0.067	0.038	0.048
Q11	E11	H11	Server	0.297	0.562	0.453	0.502
Q12	E12	H12	Server	0.090	1.000	1.000	1.000
Q13	E13	H13	TM	0.072	0.125	0.089	0.104
Q14	E14	H14	TM	0.079	0.036	0.027	0.031

Table 4. [Cont.].

Query	English query	Hindi query	Source	Time [sec]	Precision	Recall	F-Score
Q15	E15	H15	Server	0.094	1.000	0.046	0.088
Q16	E16	H16	Server	0.169	0.500	0.048	0.088
Q17	E17	H17	TM	0.230	0.333	0.029	0.053
Q18	E18	H18	TM	0.288	0.250	0.142	0.181
Q19	E19	H19	TM	0.348	0.200	0.126	0.155
Q20	E20	H20	Server	0.412	0.167	0.096	0.122
Q21	E21	H21	Server	0.463	0.143	0.089	0.110
Q22	E22	H22	Server	0.509	0.125	0.122	0.123
Q23	E23	H23	TM	0.566	0.052	0.046	0.049
Q24	E24	H24	TM	0.080	0.067	1.000	0.125
Q25	E25	H25	Server	0.330	0.044	0.038	0.041
Q26	E26	H26	TM	0.089	0.143	0.088	0.109
Q27	E27	H27	TM	0.103	0.067	0.057	0.061
Q28	E28	H28	TM	0.293	0.052	0.051	0.051
Q29	E29	H29	TM	0.071	0.006	0.005	0.005
Q30	E30	H30	TM	0.072	0.006	0.004	0.005
Q31	E31	H31	Server	0.091	1.000	1.000	1.000
Q32	E32	H32	TM	0.146	0.500	0.246	0.330
Q33	E33	H33	Server	0.194	0.333	0.249	0.285
Q34	E34	H34	Server	0.318	0.250	0.182	0.211
Q35	E35	H35	Server	0.373	0.200	0.129	0.157
Q36	E36	H36	TM	0.076	0.250	0.153	0.190
Q37	E37	H37	TM	0.805	0.055	0.044	0.049
Q38	E38	H38	TM	0.198	0.006	0.005	0.005
Q39	E39	H39	Server	0.158	0.005	0.005	0.005
Q40	E40	H40	TM	1.259	0.028	0.014	0.019
Q41	E41	H41	TM	0.110	0.006	0.005	0.005
Q42	E42	H42	TM	0.137	0.046	0.032	0.038
Q43	E43	H43	TM	0.068	0.006	0.005	0.006
Q44	E44	H44	Server	0.716	0.032	0.028	0.030
Q45	E45	H45	TM	0.133	0.006	0.003	0.004
Q46	E46	H46	Server	0.155	0.005	0.004	0.004
Q47	E47	H47	TM	0.087	0.006	0.005	0.005
Q48	E48	H48	Server	1.370	1.000	1.000	1.000
Q49	E49	H49	TM	0.055	0.521	0.453	0.485
Q50	E50	H50	TM	0.052	0.825	0.687	0.750

in the source field it is shown as “TM”; otherwise it is translated by SMT and shown as “Server”. In “time field”, the time required for translation of that query is presented in seconds. Then precision, recall and F-Score of that respective query are calculated. Graphs of precision, recall, F-Score and accuracy for all experiments are shown in Figs. 3a–3f.

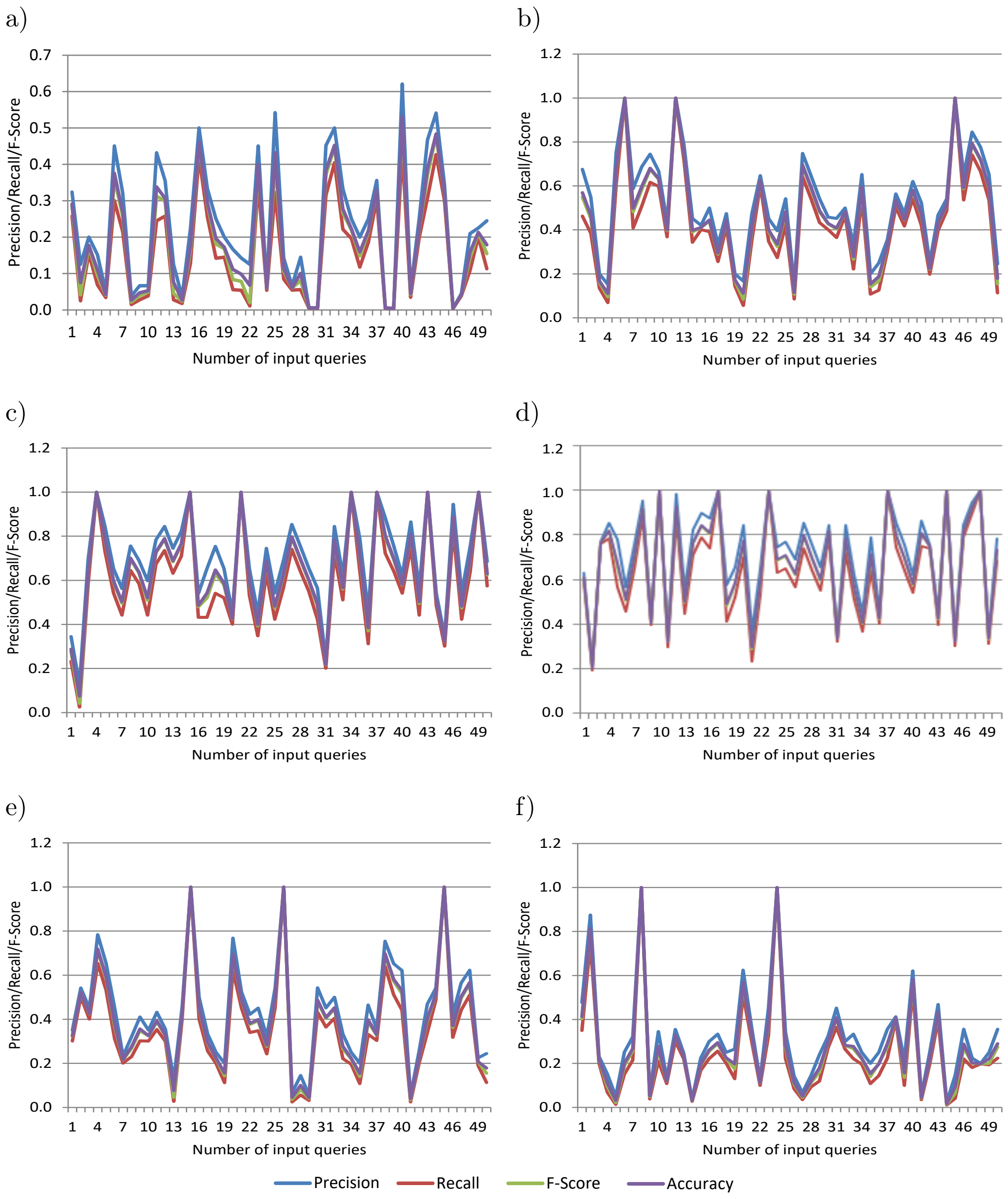


Fig. 3. Precision, recall, F-Score and accuracy graph for a) experiment 1, b) experiment 2, c) experiment 3, d) experiment 4, e) experiment 5, and f) experiment 6.

In the first experiment, training data is significantly smaller so the system performance is decreased. Then, training data increases by some amount in each next experiment and the system performance also increases. Among all the carried out experiments, experiment no. 4, in which 10 693 sentences (70%) are considered for training and 4583 sentences (30%) are considered for testing, provides a better performance. While after this, in experiments 5 and 6, when again training data is increased, the system performance is degraded. The reason is overtraining. Graphical representation of all the experiments is shown in Figs. 3a–f.

6.2. Performance test of a system in terms of time

In this research, two elements are used to enhance the MT speed of MT. One is the TM component and the second is the Hadoop (HD) infrastructure. Four experiments are carried out to check the performance of the system in terms of execution time. In each experiment, fifty queries are translated and the time required for the translation of each query is shown in the form of graph (Figs. 4–8).

Experiment 1: MT without TM and HD

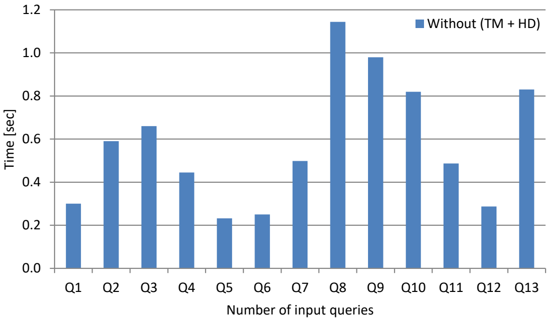


Fig. 4. The time requirement of each query without using TM and HD.

Experiment 2: MT with TM but without HD

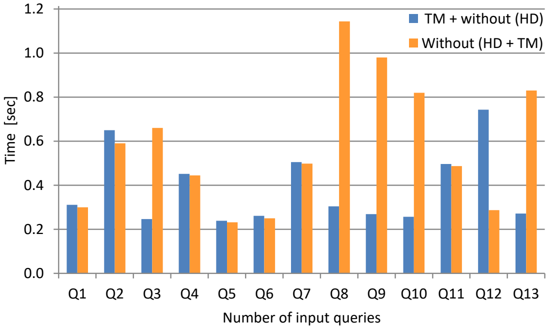


Fig. 5. Comparison of the time requirement of each query without TM and HD and with TM but no HD.

Experiment 3: MT with HD but without TM

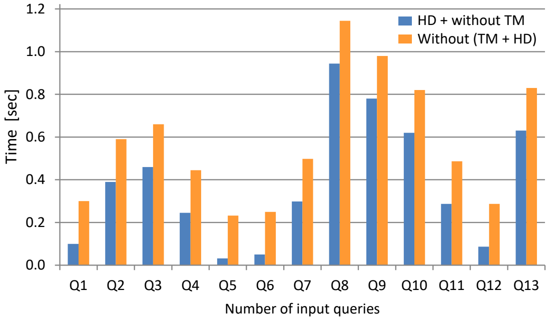


FIG. 6. Comparison of the time requirement of each query without TM and HD and with HD but no TM.

Experiment 4: MT with (TM and HD)

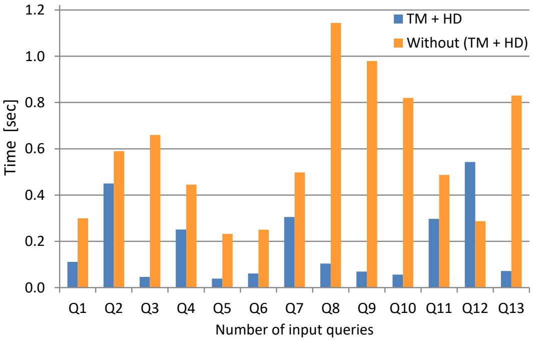


FIG. 7. Comparison of the time requirement of each query without TM and HD and with TM and HD.

Analysis of all the experiments with and without TM and HD

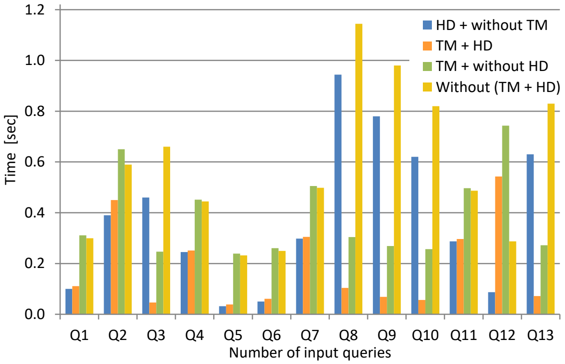


FIG. 8. Comparison of the time requirement of each query without TM and HD, with TM but no HD, with HD but no TM, and with (TM and HD).

In experiment 1, translations are carried out without using TM and HD. All the fifty queries are translated by the system. In these fifty queries. some queries are given repeatedly but take the same time to translate, as the translation process goes through all the steps every time. The total time required for translation in experiment 1 is 28.81 seconds. Graph representing the time requirement of each query without TM and HD is shown in Fig. 4. In experiment 2, translations are carried out with TM but without HD. In this case, when some queries are being repeated, they will not be translated by the system but they will be fetched directly from TM, which saves the translation time. Therefore, in experiment 2, the time for translating fifty queries is 21.53 seconds, which is less compared to the first experiment. Comparison graph of the time requirement of each query without TM and HD and with TM but no HD is shown in Fig. 5. In experiment 3, translations are carried out with HD and without TM. In this case, the system is ported on the Hadoop infrastructure, so the translation time is reduced. It takes a total of 18.81 seconds for translation. Comparison graph of the time requirement of each query without TM and HD and with HD but no TM is shown in Fig. 6. In experiment 4, translations are carried out with TM and HD. As both the components are used for improving the translation speed, the time required for translation is 11.53 seconds. A comparison graph of the time requirement of each query without TM and HD and with TM and HD is shown in Fig. 7. Finally, we have compared all the experiments and the comparison graph of all the experiments carried out is shown in Fig. 8. It represents the time requirement of each query without TM and HD, with TM but no HD, with HD but no TM, and with TM and HD.

7. Comparative analysis

The system is evaluated by finding precision, recall and F-Score after performing some experiments. This research work is compared with the results in [19], and the values of precision, recall and F-Score of the latter one are listed in Table 5. Rani *et al.* [19] used a semi-supervised method to disambiguate the sense of the given text. The performance of our system is satisfactory. It can generate better results if the size of the dataset increases.

Table 5. Comparative analysis.

System	Precision	Recall	F-Score
(Our system) Supervised system queries are from corpus	76.43	72.65	74.49
(Our system) Supervised system and queries are out of corpus	25.97	20.54	20.55
Compared system [19] Semi-supervised system and queries are out of corpus	28.93	22.90	25.56

8. Conclusions

Language is the best source to express our thoughts. Every language has some wisdom stored in it that requires investigation. This research work discussed an MT technique that automatically translates English text into Hindi by applying certain conventions. Nowadays, statistical methods are very popular and give remarkable results in MT. In this research, we translated the given English sentence into Hindi using SMT, where Giza++ does alignment. To speed up the translation process, the concept of TM is used. The system performance is improved by using SMT and TM. In the future, we would like to work in the direction of using MT in other Indian languages such as Marathi, Telugu, and Gujarati. We would also like to work in other areas such as agriculture, politics and entertainment. Proposed work can be applied to Marathi, Telugu or Gujarati. The only thing required is parallel corpus in that language. Once the corpus is available, we can apply the same phases to obtain the results.

References

1. D. Pinto, D. Vilariño, C. Balderas, M. Tovar, B. Beltrán, Evaluating n-gram models for a bilingual word sense disambiguation task, *Computación y Sistemas*, **15**(2): 209–220, 2011.

2. M. Artetxe, G. Labaka, E. Agirre, Unsupervised statistical machine translation, [in:] *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Brussels, Belgium, pp. 3632–3642, 2018, doi: 10.18653/v1/D18-1399.

3. G. K. Sidhu, N. Kaur, Role of machine translation and word sense disambiguation in natural language processing, *IOSR Journal of Computer Engineering (IOSR-JCE)*, **11**(3): 78–83, 2013.

4. N. Sharma, P. Bhatia, English to Hindi Statistical Machine Translation, *International Journal of Advances in Computer Networks and Its Security*, **1**(1): 362–366, 2011.

5. Y.S. Chan, H.T. Ng, D. Chiang, Word sense disambiguation improves statistical machine translation, [in:] *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 33–40, Prague, Czech Republic, 2007.

6. J. Cho, W. Huh, Unsupervised word sense disambiguation using association rules from XML document, *International Journal of Applied Engineering Research IJAER*, **9**(24): 29609–29616, 2014.

7. J.G. Cho, K.C. Shin, A graph-based word sense disambiguation using measures of graph connectivity, *KIIT*, **12**(6): 143–152, 2014, doi: 10.14801/kiitr.2014.12.6.143.

8. P. Desai, A. Sangodkar, Om P. Damani, A domain-restricted, rule based, English-Hindi machine translation system based on dependency parsing, [in:] *Proceedings of the 11th International Conference on Natural Language Processing (ICON)*, Goa, India, 2014.

9. A.R. Pal, D. Saha, Word sense disambiguation in Bengali language using unsupervised methodology with modifications, *Sādhanā,* **44,** Article no. 168, Indian Academy of Sciences, 2019, doi: 10.1007/s12046-019-1149-2.

10. A. Fraser, D. Marcu, Getting the structure right for word alignment: LEAF, [in:] *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Association for Computational Linguistics, Prague, Czech Republic, pp. 50–60, 2007.

11. Y. Jiang, W. Han, K. Tu, A regularization-based framework for bilingual grammar Induction, [in:] *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1423–1428, Hong Kong, China, 2019, doi: 10.18653/v1/D19-1148.

12. R. Mahendra, H. Septiantri, H.A. Wibowo, R. Manurung, M. Adriani, Cross-lingual and supervised learning approach for Indonesian word sense disambiguation task, [in:] *Proceedings of the 9th Global WordNet Conference – GWC 2018*, pp. 245–250, Nanyang Technological University (NTU), Singapore, 2018.

13. Y. Xia, Research on statistical machine translation model based on deep neural network, *Computing*, **102**: 643–661, 2020, doi: 10.1007/s00607-019-00752-1.

14. S. Melacci, A. Globo, L. Rigutini, Enhancing modern supervised word sense disambiguation models by semantic lexical resources, [in:] *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan, pp. 1012–1017, 2018.

15. D. Melamed, A word-to-word model of translational equivalence, [in:] *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, Madrid, Spain, July 1997, pp. 490–497, 1997, doi: 10.3115/976909.979680.

16. S. Štajner, M. Franco-Salvador, S.P. Ponzetto, P. Rosso, H. Stuckenschmidt, Sentence alignment methods for improving text simplification systems, [in:] *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pp. 97–102, Vancouver, Canada, July 30 – August 4, 2017, doi: 10.18653/v1/P17-2016.

17. N. Pourdamghani, M. Ghazvininejad, K. Knight, Using word vectors to improve word alignments for low resource machine translation, [in:] *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol. 2 (Short Papers), pp. 524–528, 2018, doi: 10.18653/v1/N18-2083.

18. X. Pu, N. Pappas, J. Henderson, A. Popescu-Belis, Integrating weakly supervised word sense disambiguation into neural machine translation, *Transactions of the Association for Computational Linguistics*, **6**: 635–650, 2018, doi: 10.1162/tacl_a_00242.

19. P. Rani, V. Pudi, D. Sharma, Semisupervised data driven word sense disambiguation for resource-poor languages, [in:] *Proceedings of the 14th International Conference on Natural Language Processing (ICON 2017)*, pp. 503–512, Kolkata, India, 2017.

20. A. Saif, N. Omar, U.Z. Zainodin, M.J. Ab-Aziz, Building sense tagged corpus using Wikipedia for supervised word sense disambiguation, *Procedia Computer Science*, **123**: 403–412, 2018, doi: 10.1016/j.procs.2018.01.062.

21. A.R. Shahid, D. Kazakov, Using parallel corpora for word sense disambiguation, [in:] *Proceedings of Recent Advances in Natural Language Processing*, Hissar, Bulgaria, pp. 336–341, 2013.

22. N. Sharma, *English to Hindi statistical machine translation system*, Master thesis, Thapar University, Patiala, India, 2011, https://tudr.thapar.edu:8443/jspui/handle/10266/1449.

23. K. Neerajaa, R.B. Padmaja, K. Srinivas Rao, Graph-based word sense disambiguation in Telugu language, *International Journal of Knowledge-based and Intelligent Engineering Systems*, **23**(1): 55–60, 2019, doi: 10.3233/KES-190399.

24. A.V. Subalalitha, B. S. Baqui, Statistical machine translation from English to Hindi, *International Journal of Pure and Applied Mathematics*, **118**(20): 1649–1655, 2018.

25. A.M. Bigvand, T. Bu, A. Sarkar, Joint prediction of word alignment with alignment types, *Transactions of the Association for Computational Linguistics*, **5**: 501–514, 2017, doi: 10.1162/tacl_a_00076.

26. X. Wang, Z. Tu, M. Zhang, Incorporating statistical machine translation word knowledge into neural machine translation, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **26**(12): 2255–2266, 2018, doi: 10.1109/TASLP.2018.2860287.

27. S. Yamaki, H. Shinnou, K. Komiya, M. Sasaki, Supervised word sense disambiguation with sentences similarities from context word embeddings, [in:] *Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation (PACLIC 30)*, October 2016, Seoul, South Korea, pp. 115–121, 2016.

28. IIT Bombay English-Hindi Corps, http://www.cfilt.iitb.ac.in/wsd/annotated_corpus/, last accessed on 05.11.2018.

29. B. Moradi, E. Ansari, Z. Žabokrtsky, Unsupervised word sense disambiguation using word embeddings, [in:] *Proceedings of the 25th Conference of Open Innovations Association (FRUCT)*, 5–8 Nov., 2019, Helsinki, Finland, 2019.

30. A. Kumari, D.K. Lobiyal, Efficient estimation of Hindi WSD with distributed word representation in vector space, *Journal of King Saud University Computer and Information Sciences*, [in press] 2021.

31. S. Rawat, Supervised word sense disambiguation using decision tree, *International Journal of Recent Technology and Engineering (IJRTE)*, **8**(2): 4043–4047, 2019.

32. S.G. Rawat, M. B. Chandak, N.A. Chavan, An approach for improving accuracy of machine translation using WSD and GIZA, *International Journal of Computer Sciences and Engineering*, **5**(10): 256–259, Oct. 2017, doi: 10.26438/ijcse/v5i10.256259.

33. S. Rawat, M.B. Chandak, A. Chavan, An approach for efficient machine translation using translation memory, [in:] A. Unal, M. Nayak, D.K. Mishra, D. Singh, A. Joshi [Eds.], *Smart Trends in Information Technology and Computer Communications. SmartCom 2016. Communications in Computer and Information Science*, vol. 628, Springer, Singapore, doi: 10.1007/978-981-10-3433-6_34.

34. S. Rawat, M. Chandak, Comparative survey of document analysis and categorization techniques, [in:] *Proc. of the International Conference On Recent Advances in Computer Science, E-Learning, Information & Communication Technology (CSIT – 2016)*, New Delhi, India, **3**(1): 37–41, 2016.

35. S. Rawat, A review on word sense disambiguation, *International Journal of Innovative Research in Computer & Communications Engineering*, **3**(4): 2750–2755, April 2015, doi: 10.15680/ijircce.2015.0304012.

36. S. Rawat, A comparative study on different approaches to word sense disambiguation, [in:] *Proceedings of the National Conference on Research in Cloud and Cyber Security (NCRCCS 2015)*, Nagpur, India, 2015.

37. S. Rawat, M. Chandak, Word sense disambiguation and classification algorithms: A review, *International Journal of Computer Science and Applications* (Proc. of NCRMC-2014, RCoEM, Nagpur, India as a Special Issue of IJCSA), **8**(1): 4–8, 2015.

Received September 17, 2021; revised version February 9, 2022; accepted March 15, 2022.