Word Sense Disambiguation in Hindi Language Using Score Based Modified Lesk Algorithm

Prafullit Tripathi¹, Prasenjit Mukherjee², Manik Hendre³, Manish Godse⁴, Baisakhi Chakraborty⁵

¹,⁵Department of Computer Science and Engineering, NIT, Durgapur, India
²,³,⁴Department of Analytics and IT, PIBM, Pune, India

¹ragapt1507@gmail.com, ²prasen.mscit09@gmail.com, ³manikcom12@gmail.com, ⁴manish.godse@pibm.in,
⁵baisakhichak@yahoo.co.in

Abstract: Hindi is the widely used spoken language in the Indian subcontinent, and is used by more than 260 million Indian citizens. Indian governments have many digital initiatives to serve Indian citizens better, hence Hindi language becomes one of the important languages to serve Indian citizens. The Government initiatives are like smart city, Hospital Services, Common Service Centers, Digital Payment Ecosystem, Pensioners Scheme, Digital Locker and many more. These all initiatives are served using mobile and web based applications, which citizens can access easily instead of visiting various government departments. To serve the large Hindi speaking population, it is necessary to handle the ambiguous words which have multiple connotations in any natural language processing task. In this paper, word sense disambiguation for Hindi language is proposed. Proposed method makes use of Lesk algorithm to disambiguate the Hindi words. Novel scoring method is used to assign a sense score to each token of the Hindi sentence. The sense score is calculated based on the gloss, hypernym, hyponym and synonym of the combinations of different sense of tokens. Hindi WordNet database created by CFILT, IIT Bombay is used in the proposed system. The proposed algorithm takes a natural language (NL) sentence in Hindi (Devanagari script) and process the sentence according to the score based approach modeled on the basic Lesk algorithm with the help of Hindi WordNet designed by CFILT IIT Bombay. The solution provided in this paper can be used vividly in various web based applications like Query-Response Systems, Question-Answer Systems, Sentiment analysis, Recommendation systems etc.

Keywords: NLP, Lesk Algorithm, word sense disambiguation, multi word WSD, Hindi WordNet.

1. INTRODUCTION

Hindi is a Devanagari scripting language. Most of the citizens in the India and its subcontinent uses Hindi language for communication. After Digital India initiative, demand of regional language based web applications are increasing exponentially. Citizens are using application in Healthcare, Transportation, Education, Tourism, Financial and Pension Applications which are in regional languages. Citizens are flexible with their native language and Hindi is the most acceptable language in India and its subcontinent. Native speaker of Hindi is almost 260 million people according to Ethnologue as in [1]. Many Hindi based applications have been developed that are related to the Query-Response Systems, Question-Answer Systems, Sentiment analysis systems, etc. These systems are natural language processing (NLP) based systems, hence has the problem of Word Sense Disambiguation. Understanding of a sense in a Sentence or Word by the NLP based systems is an open research problem. Human languages contain many ambiguous words that create problems at the time of processing sentences. Human languages contain vocabulary that is comprised of words. All the languages are full of words that have multiple meanings associated with it. The meaning of a context is dissected depending on the words used in the sentence. Sometimes same word can have multiple senses [2]. These words are called polysemous words. A sentence with one or more polysemous words in it is said to be ambiguous. Human neural networks (human brain) are way too efficient in disambiguating the correct sense of a word used in a sentence. However, computer systems are not too proficient in it. Human languages are too complex for machines to interpret and understand. It is preferred not to have such ambiguities in computer related applications. Identification of polysemous words and assigning their correct sense is classical problem in NLP.
Process of assigning senses to the polysemous words is known as word-sense disambiguation problem.

Word sense disambiguation is one of the major areas of interest for NLP researchers. Computer programs do not have the human experience in languages, so automatically correct sense detection of a polysemous word is a difficult task. This is the problem that Word Sense Disambiguation (WSD) tries to solve. WSD is a core research problem in NLP which tries to identify the sense of a word which is used in a given context as in [3] [4]. Most of the WSD algorithms [5] [6] are divided as supervised, unsupervised, and knowledge-based techniques and that can be utilized to improve many NLP based applications like Information Retrieval [7] [8], Sentiment Analysis [9] [10], Machine Translation [11] [12] or Text Summarization [13] [14]. Supervised learning [15] requires large volumes of human tagged data in order to find set of rules to disambiguate future queries. There are two drawbacks to this approach. First, human interaction is required. Second, no set of rules can always dissect correct meaning of the word. On the other hand, unsupervised learning [16] uses classical sources like dictionary based WordNet [17] etc. to disambiguate senses. Lesk algorithm falls under the category of knowledge base based algorithm. Classical Lesk algorithm offers a score based approach to find overlap between different senses of target word and words nearby to it. The Lesk algorithm is one of the most well-known algorithm for word sense disambiguation which is introduced by Michael E. Lesk [2] in 1986. A simplified version of the Lesk algorithm is to compare the dictionary definition of an ambiguous word with the words contained in its neighborhood. Versions have been adapted to use WordNet. The proposed work presents an adaptation of Lesk algorithm that includes dictionary to disambiguate ambiguous Hindi words from a Hindi sentence. Rather than using a standard dictionary, this algorithm employs a different kind of dictionary called as WordNet. Traditional dictionary arranges words alphabetically while WordNet organize the words semantically. Glosses refer to the definitive meaning of the word. Overlapping between the glosses of the neighbouring words plays a key role in Lesk’s approach [2]. The gloss of a word provides us the following information.

1. List of different senses of the word in WordNet. Each sense belongs to a different synset.
2. Information about the context in which word appears.

A scoring method is provided with the algorithm in order to give sense score to each set of combination. This scoring method is carried out by considering all pairwise matching such that each element of the pair belong to the same detail of separated words. For a given query having ‘n’ tokens each having ‘k’ senses, there will be total ‘nk’ different combinations. Proposed scoring method is applied to all these different combinations to give sense scoring and choose appropriate sense. Before applying word sense-count firstly, we have to trim the query so as to reduce the useless words (stop words) for unnecessary computation. Further, we develop the query by searching stemmed words for validity. If there is no such validity in WordNet, we reject the word just like stops words. After a data set generation, we implement the Lesk algorithm for further processing of sentences. Word sense disambiguation is an open problem in computational linguistic where it detects correct sense of a word that is used in a sentence. In the Simplified Lesk algorithm, the meaning of a word is calculated by finding the overlaps between the context and the dictionary meanings of the word. The meaning for which maximum overlap takes place is taken as the word sense. The word sense of each word is found individually. Internet is emerging as a major source of communication in India. It is expected to have large data analysis requirement in regional local languages with the availability of internet services is rural areas. Hindi language contains many ambiguous words handling which is difficult for NLP systems because meaning of a same word is different in different sentences with different contexts.

In this paper, we have proposed to modify the Lesk Algorithm to disambiguate ambiguous Hindi words efficiently. The score and combination based approach have been applied on Lesk algorithm. NLP based steps like tokenization, lemmatization, POS tagging are used to process the Hindi sentences. The Hindi WordNet is used to extract the correct sense of an ambiguous Hindi word. The International Phonetic Alphabet (IPA) notation [1] for Hindi words is used in this research work. The proposed algorithm takes Hindi sentences as an input. The output will be generated as correct sense of the Hindi word. Many WSD systems have been discussed briefly in Related Works Section (Section 2). The Architecture of this WSD system has been elaborated in Section 3. The Time Complexity of Proposed system has been calculated in Section 4 where Methodology has been given in Section 5. Applications of the system have been stated in Section 6. Finally, the Future Work and Conclusion has been drawn in Section 7 and Section 8 respectively.

2. RELATED WORKS

Word sense disambiguation is one of the key problems in the NLP domain. A large volume of research is going on
for the selection of an efficient WSD algorithm. Majority of the unsupervised learning solutions use Lesk Algorithm as their underlying algorithm. Basic Lesk algorithm is implemented for the English corpus. Some of the languages does not have rich corpus like English. Such languages require some variations in classical Lesk algorithm in order to disambiguate senses efficiently. Rajat Pandit proposed dependency tree based lesk variant to perform WSD for low resource language like Bengali [18]. Shancheng Tang proposed a solution using deep Chinese word sense disambiguation method which is based on sequence to sequence that improved performance by 11.48% [19]. Evaluation of different WSD algorithm is a major sub problem in the NLP domain. Alessandro Raganato designed a unified evaluation system and measure the performance of various word sense disambiguation algorithms. This paper offers two main contributions. The first contribution is (1) standardizing the WSD datasets and training corpora into a structured format, (2) semi-automatically annotation conversion from any dataset to WordNet 3.0, and (3) pre-processing the datasets by consistently using the same pipeline. Second, the evaluation framework has been used to perform a fair quantitative and qualitative empirical comparison of the main techniques that is discussed in the WSD literature, including the latest advances based on neural networks [20]. Quang-Phuoc Nguyen designed a lexical semantic network for Korean language which is useful for Korean morphological and word sense disambiguation [21]. The Lexical semantic network [21] uses largest Korean LSN that consist of Lexical networks of nouns, predicates, and adverbs. Each node is connected with other node using six types semantic relation that are hyponymy, synonymy, similarity, antonym, part-whole, and association relations. To represent a certain sense of a word, each node is consisted with a word and its sense code. Korean morphological analysis is difficult because different types of morphems and parts of speech are encoded into the same eojeol. The meaning of a morphem can be changed because of different parts of speech tagging with the morphems. The morphological analysis refers to discover the correct set of sense in a given context. In Korean morphological analysis, the conventional methods are a) Segment the input eojeol into morphemes and b) tag POS to each morpheme as in [21]. The application of WSD is not limited to any particular domain. Saeed Seifollahi, Mehdi Shajari use WSD to analyze the sentiments of news headlines in order to predict their impact on the stock prices. Some of the applications require heavy computations. To make the application time efficient, several heuristics can be utilized to improve the performance. Genetic algorithms are used widely to address various hard optimization problems. Zankhana B. Vaishnav applied Knowledge-Based Approach to disambiguate polysemous words Using Genetic Algorithm. Genetic approach proposed used Indo-Aryan WordNet for Gujarati language as lexical database as in [23]. S. N. Mohan Raj propose method to eliminate ambiguity due to homonymy using cluster and deep learning approach. Homonymy ambiguity is a problem in Malayalam language. This problem is attempted by the Authors in [24]. The system uses POS tagging lemmatization and sense annotation. The neural network has been used in deep learning method and this method is using the corpus to disambiguate the homonymous words in Malayalam as in [24]. Jagbir Singh proposed a word Sense disambiguation system that is based on Punjabi language. The enhanced Lesk approach has been utilized to extract correct sense of ambiguous Punjabi words. The methodology of this system is supervised learning and Indo WordNet has been used for disambiguating Punjabi words as in [25]. Lekshmi R Pillai proposed a question answering system to predict an answer for an input question. This model is based on a combined approach using word sense disambiguation (WSD) and semantic role labelling (SRL). The proposed system is a factoid sense based question generation system. The Lesk tool has been used for WSD where the senna tool has been used for SRL. They used Lesk tool for WSD and senna tool for SRL which are based on the sense affiliated with the sentence system generates questions that are semantically solvable as in [26]. Word sense disambiguation can be solved using three kinds of approaches, knowledge based, corpus based and hybrid approach. Himdweep Walia proposed a Naïve Bayes based WSD approach that has shown higher accuracy than other implementations as in [27]. Training data is an issue to disambiguate ambiguous words in Persian language that has been faced by the Author [28]. The Machine Learning (ML) algorithm with minimum supervision has been considered to solve this problem. Various news articles have been used as main source of the reference corpus. This method uses some predefined features of target words to disambiguate senses of the word as in [28]. G. Sajini designs a user interface where user manually enters a sentence in Hindi with polyeomy word. This system [29] identifies the polysemous words and list one or more meanings that are associated with that word. Using machine learning techniques, this system identifies the correct meaning of the polysemous word which is based on the given context as in [29]. Author [30] proposed a WSD system which is based on Naive Bayes classifier (ML technique). Sense corpus and Ambiguous corpus have been used by this WSD system.
TABLE I.  COMPARATIVE STUDY AMONG SIMILAR TYPE WSD SYSTEMS WITH THE PROPOSED WSD SYSTEMS

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Authors</th>
<th>WSD based Systems</th>
<th>Technique(s) used in similar type WSD System</th>
<th>Technique(s) used in proposed WSD System</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Rajat Pandit et al. [18]</td>
<td>Word Sense Disambiguation by Improvised Lesk Algorithm [18]</td>
<td>The LESK algorithm has been improved for Word sense disambiguation (WSD) on Bengali language. The dependency tree based Lesk has been used to perform the WSD.</td>
<td>The classical Lesk algorithm has been modified for the WSD in Hindi language. The proposed algorithm uses score based approach.</td>
</tr>
<tr>
<td>2</td>
<td>Shancheng Tang et al. [19]</td>
<td>Deep Chinese Word Sense Disambiguation Method [19]</td>
<td>Chinese WSD method has been used in this system. The Deep learning method has been used for feature extraction of Chinese text.</td>
<td>The Hindi sentence is used to process according to the score which is modeled on basic Lesk algorithm. The score is assigned on each combination of sense. The largest sense score is the output of this system.</td>
</tr>
<tr>
<td>3</td>
<td>Alessandro Raganato et al. [20]</td>
<td>A Unified Evaluation Framework on Word Sense Disambiguation [20]</td>
<td>An evaluation framework has been designed, that is based on various WSD. The first step of evaluation is standardizing the WSD datasets and training corpora into a unified format, The second step is Converting annotation from any dataset to WordNet and third step is pre-processing of dataset using same pipeline. The proposed frame work performs a comparison which based on quantitative and qualitative.</td>
<td>The proposed system is using supervised method of Lesk algorithm. The Hindi WordNet of CFILT, IIT, Bombay has been used to implement this system. This system is able disambiguate ambiguous Hindi words easily.</td>
</tr>
<tr>
<td>4</td>
<td>Quang-Phuoc Nguyen et al. [21]</td>
<td>Korean-Vietnamese Neural Machine Translation System [21]</td>
<td>Korean is a morphologically rich language where Vietnamese is an analytical language. Korean language contains word ambiguities which is a problem for Neural MT. The Korean knowledge base has been prepared for the lexical semantic network which is used for morphological analysis and WSD in Korean language. Another large Korean-Vietnamese corpus has been prepared for the Vietnamese word segmentation method.</td>
<td>Hindi is an Indo-Aryan language. Hindi is used by most of the people in India and Indian subcontinent. Understanding a Hindi Sentence is a real issue because Hindi sentence contains many ambiguous Hindi words. This problem has been solved by the Lesk algorithm using Hindi WordNet. The Classical Lesk algorithm has been modified.</td>
</tr>
<tr>
<td>5</td>
<td>Saeed Seifollahi et al. [22]</td>
<td>Word Sense Disambiguation in Sentiment Analysis [22]</td>
<td>A sentiment analysis has been done on the News headlines. The WSD has been used on sentiment analysis to predict the impact of the stock price. They have used the news headlines as input of the proposed system.</td>
<td>The Score based approach has been applied on Lesk algorithm to disambiguate the ambiguous words. A sense score has been assigned to each possible combination of senses where largest sense score is the output of this proposed system.</td>
</tr>
<tr>
<td>6</td>
<td>Zankhana B. Vaishnav et al. [23]</td>
<td>Word Sense Disambiguation in Gujarati Language [23]</td>
<td>A genetic algorithm has been used to Disambiguate polysemous words. The proposed system is based on the Gujarati language. It uses Indo-Aryan WordNet database for the Gujarati as lexical database.</td>
<td>A score based modified Lesk algorithm has been used to disambiguate the ambiguous words. The proposed system is based on Hindi language.</td>
</tr>
<tr>
<td>7</td>
<td>S. N. Mohan Raj et al. [24]</td>
<td>Word Sense Disambiguation of Malayalam Nouns [24]</td>
<td>This system uses WSD on Malayalam language. The proposed work concerned about the homonymy ambiguity. To resolve the homonymy ambiguity, two approaches have been given- Clustering and Deep learning. Clustering is supervised method where POS tagging, lemmatization and sense annotation have been used. The Deep learning approach is based on neural network that uses corpus for disambiguating homonymous words.</td>
<td>The proposed system uses Hindi words for disambiguation. The Hindi language is Devanagari Script and this script is containing many ambiguous words. To extract the correct sense, The Lesk algorithm has been applied after modification. Stop words elimination, Stemming, POS tagging, Sense generation are most popular steps have been applied on this proposed system.</td>
</tr>
<tr>
<td>No.</td>
<td>Authors</td>
<td>Title</td>
<td>Description</td>
<td>Source</td>
</tr>
<tr>
<td>-----</td>
<td>----------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>8</td>
<td>Jagbir Singh et al. [25]</td>
<td>Word Sense Disambiguation in Punjabi [25]</td>
<td>This work is analyzing the correct meaning of the ambiguous words in Punjabi language. An enhanced Lesk approach have been utilized to extract the correct sense of the ambiguous words. The supervised learning methodology and Indo WordNet have been utilized to develop this system.</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Lekshmi R Pillai et al. [26]</td>
<td>Word Sense Disambiguation for Question-Answering System [26]</td>
<td>This is a factoid sense based question-answering system. The WSD and Semantic Role Labelling (SRL) approaches have been applied on this system. The Lesk is used for WSD and Senna is used for SRL.</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Himdweep Walia et al. [27]</td>
<td>Gurmukhi Word Sense Disambiguation [27]</td>
<td>Three kind of approaches is used in WSD. The first approach is knowledge base, the second approach is corpus based, and third approach is hybrid based. The Naïve Bayes classification algorithm is used on WSD to disambiguate Gurmukhi words. The Punjabi corpus have been used for Gurmukhi words.</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Mohamadreza Mahmood et al. [28]</td>
<td>Persian word sense disambiguation [28]</td>
<td>Persian language based WSD problem has been solved by the Machine Learning (ML) algorithm where minimal supervision has been considered. The corpus has been represented from various news articles. The proposed system uses some predefined features of targeted words and collaborative learning method.</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Saiba Nazah et al. [32]</td>
<td>Word Sense Disambiguation of Bangla sentences [32]</td>
<td>This WSD system is based on Bengali words. Naïve Bayes classifier and artificial neural network (ANN) have been used to disambiguate the Bengali Words. The Naïve Bayes classifier has been use in training phase and ANN has been used to detect the correct sense of the word.</td>
<td></td>
</tr>
</tbody>
</table>
The Sense corpus contains synsets, synonyms and antonyms where Ambiguous corpus contains all possible ambiguous words. The experimental result of this system is promising to disambiguate the ambiguous words as in [30].

Unsupervised approach is a most popular approach in Machine Learning. A WSD system has been described by the Author [30] which is unsupervised approach and supervised approach has been bypassed by this system using simulation of the semantic inference process that is performed by human language users as in [31]. Another Bengali WSD system has been proposed by the Author [32] that uses Naïve Bayes classifier and Artificial Neural Network (ANN) to disambiguate the ambiguous Bengali words in Bengali Language. Naïve Bayes classifier has been used for the training phase and ANN has been utilized to predict and detect the correct sense of an ambiguous Bengali word as in [32]. Adapted Lesk method has been used by the Author [33] in their WSD system. The WordNet has been utilized instead of a standard dictionary. The WordNet contains a hierarchy of a semantic relation. The proposed system takes a single target word and its surrounding words as Input. The output is generated as sense of the target word. The comparison has been done on glosses of involved words and related words like hypernym, hyponym, holonym, meronym, troponym, and attribute that are available in WordNet. The sample data from SENSEVAL-2 has been utilized for method evaluation and this system has achieved 32% overall accuracy as in [33]. A comparative study has been done on similar type WSD system and proposed WSD system. The comparative study has been given in Table 1.

3. ARCHITECTURE OF THE PROPOSED SYSTEM

Proposed System architecture (Figure-1) takes a natural language sentence in Hindi as input. It first removes Stop Words from the sentence. Stemming and POS tagging is major step for any natural language application. After eliminating the stop words from the input sentence, refined query is stemmed word by word. Further, refined query is POS tagged and only tokens with multiple senses are left. Each of these tokens are queries in the WordNet semantic database and all the information usable in future is separated in a distinct variable named as dataset. Different possible sense combinations are generated one by one. Each of the combination is assigned a value evaluated using proposed scoring method. The sense combination with maximum sense score is output of the algorithm. Output is returned to the client in the form of word sense pairs.

The proposed algorithm is modularized into following 6 steps. Each of these modules are explained in details in subsequent sections.

1. Post NL Sentence in Hindi
2. Stop Word Elimination
3. Stemming and POS Tagging
4. Dataset Generation
5. Modified Lesk Algorithm Implementation
   a. Sense Combination Generation
   b. Sense Score Assignment
6. Output (Word Sense Pair)

![Figure 1. Architecture of the Proposed WSD System](image)

3.1. Algorithm

Basic Algorithmic Steps of this Proposed WSD System has been described here:

i. Post NL Sentence in Hindi

The proposed WSD system will read natural language sentence in Hindi. It will check for sentence validation.
If sentence is not valid then system will prompt to user. Otherwise the sentence will go for next step.

ii. Stop Words Elimination

Each language has a variety of stop words which is of no use while processing them for natural language processing applications. So it would be better to eliminate these stop words prior to advance of further processing.

A few example of Stop Words used in Hindi language are listed here. These are in Devanagari script.

| या | है | के | ते | क्या | अपना | उसके | कहा | होती | अपने | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसको | कहे | उसक0

These words are Stop words which are not very important for the Hindi sentence processing. In this step, Stop words are eliminated from the natural language sentence. Example sentence has been given here in Figure 2 and 3.

Natural language sentence: मलकेआधारपरफ़ैसलाहुआ।

Pseudo code of Stop Words Elimination step has been given here.

```
Pseudo Code: Stop Words Elimination

# install morfessor library
import treetools
import math
from pyiwn import pyiwn

# initialize indicnlp library
from indicnlp import common

# place the directory of indic_nlp_resources
common.set_resources_path("C:\Users\Pranfull\Tripathi\spyder-psy\&indic_nlp_resources")
from indicnlp import loader
loader.load()
```

Examples:

a) **Determiner**: 'नहीं'(/niː[6]) (No), 'कुछ'(/ko[6][7]) (Few), 'हर'(/har) (Each), etc.


c) **Adverb**: 'यही' (/iː[6]) (here), 'अब' (/a[5]v[4]) (Now), etc.

d) **Preposition**: 'द्वारा' (/d[2]aːr aː) (By), 'जब' (/dʒaː) (If), etc.

e) **Wh Words**: 'कैसे' (/koː seː) (How), 'कहँ' (/kʰaː) (Where), 'क्या' (/kaː) (k[6]) (What), 'जैसे' (/dʒaː) (How), etc.

These words are Stop words which are not very important for the Hindi sentence processing. In this step, Stop words are eliminated from the natural language sentence. Example sentence has been given here in Figure 2 and 3.
iii. Stemming and POS tagging

Filtered query sentence (query without stop words) is again parsed word by word in order to stem the root word from each filtered token. Indicnlp library (https://pypi.org/project/indicnlp/) provides support to stem words using morphological analysis of their WordNet resource. While stemming, we further improve query by searching stemmed word into WordNet for any valid sense. We search this stem word as either noun, adverb, adjective or verb as all other part of speech is of no use. If stem word have no valid senses as above stated part of speech then we simply reject the word.

After stemming we get tokens as below:


The Pseudo Code of Stemming and POS Tagging has been given here:

```python
Pseudo Code: Stemming and POS Tagging
def tokenizer_and_stemmer(string):
    hindi_text_no_stop_word=tokenizer(string)
    hindi_text_morphe = hindi_text_morphe+ word +" "
    return hindi_text_no_stop_word
```

iv. Dataset Generation

After filtering and stemming the query sentence, we’re querying wordnet to filter out various senses of the tokens. A dictionary of tokens acts as the key and all the extracted information from the wordnet are assigned as the value to the tokens or the key. This algorithm requires hypernym, gloss, hyponym and synset values in further steps to evaluate sense scores.

Structure of the Dataset: DATASET[‘word’][ini]

Dataset is a dictionary, each token of the query sentence acts as key. Dataset is formed in a nested structure. Each sense of the token is further classified according to their part of speech. ini iterator uses to iterate through highest level of depth which access the meaning in the following order:

- “मत”(mt[2]), “आधार” (aːdʰ[2]aːr) and “फैसला”(pʰɔːslaː) have 5,14 and 2 possible senses. All these senses are encapsulated in variable name dataset.
TABLE II. SIGNIFICANCE OF INDICES USED IN DATASET

<table>
<thead>
<tr>
<th>Index Value</th>
<th>Represented property</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Gloss of the word itself</td>
</tr>
<tr>
<td>1</td>
<td>Gloss of the hypernym of the word</td>
</tr>
<tr>
<td>2</td>
<td>Gloss of hyponym of the word</td>
</tr>
<tr>
<td>3</td>
<td>Word synonymous to the word</td>
</tr>
<tr>
<td>4</td>
<td>Part of speech of the initial sense</td>
</tr>
</tbody>
</table>

**Pseudo Code: Dataset Generation**

```python
def dataset_generation(token_list):
    reject_words = ['', 'ने']
    dataset = {}
    for iter in token_list:
        if iter in reject_words:
            continue
        else:
            dataset[iter] = []

    # iterate for each word 4 times once as noun, verb, adj and verb respectively
    for pos_iter in range(0,5):
        if pos_iter == 0:
            syns = iwn.synsets(iter, pos=pyiwn.NOUN)
        elif pos_iter == 1:
            syns = iwn.synsets(iter+"ना", pos=pyiwn.VERB)
        elif pos_iter == 2:
            syns = iwn.synsets(iter, pos=pyiwn.VERB)
        elif pos_iter == 3:
            syns = iwn.synsets(iter, pos=pyiwn.ADJECTIVE)
        elif pos_iter == 4:
            syns = iwn.synsets(iter, pos=pyiwn.ADVERB)

        if len(syns)>0:
            flag=1
            for i in range(0,len(syns)):
                temp = []
                if flag == 1:
                    gloss_main = tokenizer(syns[i].gloss())
                    temp.append(gloss_main)
                else:
                    temp.append(syns[i].gloss())

                if pos_iter<=2:
                    hypernym = syns[i].hypernymy()
                else:
                    hypernym = []

                hypernym_str = ""
                for k2 in range(0,len(hypernym)):
                    if flag == 1:
                        gloss_hypo = tokenizer(hypernym[k2].gloss())
                        hypernym_str = hypernym_str + gloss_hypo + " || "
                    else:
                        hypernym_str = hypernym_str + hypernym[k2].gloss() + " || "

                    temp.append(hypernym_str)

            for j in syns[i].lemmas():
                syn_words = ""
                syn_words = syn_words + j.name() + " || "

                temp.append(syn_words)

            temp.append(syns[0].pos())
            dataset[iter].append(temp)

    return dataset
```

v. Modified Lesk Algorithm implementation

a) Sense Combination Generation

This algorithm generates all the different combinations of sense of the tokens as depicted in the Figure 4 and uses scoring method explained in the next segment of this section to give a sense score to each of the combination. A query with n tokens and each token having k senses have (n\(^k\)) different combinations to check.

If a query has 3 tokens (A, B and C) and these tokens have 3, 2 and 4 senses respectively. Then there are total 3x2x4 = 24 possible combinations of sense to check for.

![Figure 4. Various Possible Sense Combinations for 3 Token query](http://journals.uob.edu.bh)

According to the Figure 5, A\(_2\)B\(_2\)C\(_4\) is one such combination where 2\(^{nd}\) sense of word A, 2\(^{nd}\) sense of word B and 4\(^{th}\) sense of word C is used.
For the example query, possible combinations of senses are depicted in the Table 3.

<table>
<thead>
<tr>
<th>मत (Opinion)</th>
<th>आधार (According)</th>
<th>फैसला (Decision)</th>
<th>Sense Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>A₁</td>
<td>B₁</td>
<td>C₁</td>
<td>S₁</td>
</tr>
<tr>
<td>A₁</td>
<td>B₁</td>
<td>C₂</td>
<td>S₂</td>
</tr>
<tr>
<td>⋯</td>
<td>⋯</td>
<td>⋯</td>
<td>⋯</td>
</tr>
<tr>
<td>A₂</td>
<td>B₁</td>
<td>C₁</td>
<td>S₁₀</td>
</tr>
<tr>
<td>⋯</td>
<td>⋯</td>
<td>⋯</td>
<td>⋯</td>
</tr>
<tr>
<td>A₅</td>
<td>B₁₄</td>
<td>C₂</td>
<td>S₁₄₀</td>
</tr>
</tbody>
</table>

Where Aᵢ, Bⱼ, Cₖ represent the iᵗʰ, jᵗʰ, and kᵗʰ sense of the word मत (m[t]), आधार (aːdʰ[2]aːr), फैसला (pʰɔːsla) respectively and Sₓ is the sense score assigned to xᵗʰ combination.

b) Sense Score Assignment

In order to evaluate sense score for a given sense token combination, we follow the procedure depicted in the Figure 6. This algorithm considers all the pairwise matchings such that each element of a pair (a, b) belong to sense details of separate words. For every sense of each word gloss, hypernym, hyponym and synonym are considered to calculate a sense score for that pair.

For each pair (a, b) where a and b are both strings. We use the procedure elaborated below to assign scores.

For pair with a!=syn and b!=syn:
Common substring of length = len within a and b
Score = Score + (len * (len + 1)) / 2

Otherwise

Common substring of length = len within a and b
Score = Score + 2x len

The algorithm provides additional score in case the gloss of the one element of pair matches with the synonym directly, as it increases the probability of referred sense as the correct sense.

Pseudo Code 1: Sense Score Assignment

```python
def scoring_method(string_a, string_b, factor):
score = 0
count = 0
start_index = 0
max_count = 0
stri = ""
string_a_list = string_a.split()
string_b_list = string_b.split()
iter1 = 0
iter2 = 0
while iter1 < len(string_a_list):
    iter2 = iter2
    while len(string_a_list) > 0:
        if string_a_list[iter1] == string_b_list[iter2] and
           (string_a_list[iter1] not in common_words) and
           len(string_a_list[iter1]) > 0:
            stri = stri + string_a_list[iter1] + " ">
            count = count + 1
            iter1 = iter1 + 1
        else:
            if count > 0:
                iter2 = iter2 - 1
                iter1 = start_index
            count = 0
            stri = ""
            if count > max_count:
                max_count = count
                iter2 = iter2 + 1
if factor == 1:
    score = score + math.pow(max_count, 3)
else:
    score = score + math.pow(max_count, 2)
return score
```

Pseudo Code 2:
import itertools
from nltk import wordnet as wn
from nltk.corpus import wordnet as wn

# Define the algorithm
def lexsk_algorithm(dataset):
    f = open("out.txt", "w")
    dataset_comb = []
    temp_list = []
    for keys1 in dataset.keys():
        temp_list.append(keys1)
        dataset_comb.append(list(range(0,len(dataset[keys1]))))

    print(len(list(itertools.product(*dataset_comb))))
    print(dataset_comb)

    sense_comb = list(range(0,4*len(dataset.keys())))
    max_sense_score = 0
    factor = 0
    for list_iter in list(itertools.product(*dataset_comb)):
        sense_comb = list(range(0,4*len(dataset.keys())))
        factor = 0
        second_sense_iter = 0
        second_word_index = int(second_sense_iter/4)
        first_word_index = int(first_sense_iter/4)
        sense_result = list_iter
        max_sense_score = sense_score
        for first_sense_iter in sense_comb:
            first_sense_index = int(first_sense_iter/4)
            for second_sense_iter in sense_comb:
                second_sense_index = int(second_sense_iter/4)
                if second_sense_iter >= len(sense_comb):
                    break
                second_word_index = int(second_sense_iter/4)
                second_sense_index = int(second_sense_iter/4)
                if second_sense_index == 3 or first_sense_index == 3:
                    factor = 1
                sense_score = sense_score +
                scoring_method(dataset[temp_list[first_word_index]][list_iter[first_word_index]][first_word_index],
                                dataset[temp_list[second_word_index]][list_iter[second_word_index]][second_word_index],factor)
                i = i + 1
                if sense_score > max_sense_score:
                    max_sense_score = sense_score
                    temp_list[i] = list_iter
                    max_score = sense_score
                    sense_result = list_iter
        f.write(str(list_iter) + " " + str(max_score) + " " + str(factor) + " " + str(list_iter[0]))
    print('max_score=' + max_score + ' ' + str(factor) + ' ' + str(list_iter[0]))
    f.close()

# Function to print the results
def print_results(sense_result):
    for word in sense_result:
        print(word)

# Function to calculate the sense score
def sense_score(dataset_comb):
    pass

# Dataset generation
dataset = {'man': {'man': 'man', 'woman': 'woman'}, 'woman': {'man': 'man', 'woman': 'woman'}}

# Scoring method
scoring_method = lambda x, y, z: x + y + z

# Main function
lexsk_algorithm(dataset)

Output (Word Sense Pair)

Dataset generated is parsed with the help of scoring criterion and each of the possible outcomes of the word sense combinations is evaluated. The one combination that yields best score (maximum score) is termed as the desired output of the query. Output for the algorithm is displayed in the form of word-sense pairs where each word is matched with its corresponding sense. Words and their senses with regard to combination associated with maximum sense score are displayed as output. For the example sentence output of the algorithm is depicted in Figure 7.

Output = Sense (max( s1,s2,s3...,s140))
The word खेल (Khel, Hindi), Play (English) has 7 senses as a noun and 10 senses as verb and thus has different synsets for each of these senses in the WordNet. We also have hyponyms and hypernyms for each word.

For example the word खेल (Khel (Hindi), Play (English)) has 7 senses as a noun and 10 senses as verb and thus has different synsets for each of these senses in the WordNet. Thus 17 in total.

Assume, each of the l’ tokens having $S_1, S_2 ... S_l$ Senses associated with them.

Let, query sentence is ‘l’ words long and it contains $k_1$ Stop words and $k_2$ word with no useful senses (words with no sense as noun, verb, adverb or adjective).

Tokens after the filtering and stemming step = $1 - k_1 - k_2 = l$

This complexity can be further improved by using any efficient string matching algorithm like KMP algorithm or Robin Karp algorithm etc.

Overall complexity of the algorithm = $O (C \times L^2 \times n^2)$

5. METHODOLOGY

Each of the sub steps explained in the architecture is elaborated here with an example sentence in Hindi.

Input Hindi Sentence: “आम मेरा घर जल गया” (a am me ra ga)(a a g)

English Sentence: “My house got burnt in the fire”

Figure 8 shows the Stop word in the given example sentence. Sentence consists of “म” which is a stop word.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>आम</td>
<td>मेरा</td>
<td>घर</td>
<td>जल</td>
<td>गया</td>
<td></td>
</tr>
</tbody>
</table>

![Stop Word](http://journals.uob.edu.bh)

i) Filtered query after stop words elimination has been given in Figure 9.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>आम</td>
<td>-</td>
<td>मेरा</td>
<td>घर</td>
<td>जल</td>
<td>गया</td>
</tr>
</tbody>
</table>

![Figure 9. After Stop Word Elimination](http://journals.uob.edu.bh)
Example: For word ‘जल’ (dʒl) (Jal), root word is ‘जलाना’ (dʒlənə) (Jalanaa).
Now, the Stemmed sentence has been given in Figure 10.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>आग</td>
<td>-</td>
<td>घर</td>
<td>जल,जलाना</td>
<td>गया</td>
<td></td>
</tr>
</tbody>
</table>

Figure 10. Stemmed Sentence

iii) POS tagging identifies the part of speech associate with each token in the filtered sentence. If there is no noun, adjective, verb, adverb sense associated with the token then algorithm simply reject that token as all the words belong to remaining part of speech is unisemous that does not need any disambiguation i.e. preposition, conjunction etc. Figure 11 and Figure 12 shows the rejection of unisemous word.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>आग</td>
<td>मेरा</td>
<td>घर</td>
<td>जल,जलाना</td>
<td>गया</td>
<td></td>
</tr>
</tbody>
</table>

Figure 11. Unisemous word

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>आग</td>
<td>मेरा</td>
<td>घर</td>
<td>जल,जलाना</td>
<td>गया</td>
<td></td>
</tr>
</tbody>
</table>

Figure 12. Stemmed Sentence

iv) Dataset generation step form a dataset that include all possible sense of the token left in the filtered sentence. A component of generated dataset is depicted in the Figure 13.

<table>
<thead>
<tr>
<th>Token</th>
<th>आग</th>
<th>घर</th>
<th>जल,जलाना</th>
<th>गया</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of senses</td>
<td>3</td>
<td>10</td>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>

Figure 14. Dataset Format

According to Figure 15, Total no. of sense combinations possible are = 3 x 10 x 11 x 3 = 990 combinations.

vi) Each of the 990 combinations are evaluated and one with the maximum score assigned as the output of the algorithm.

Python has been used to implement this algorithm. Python is an open source software tool and it is available in Web with its all modules and supporting libraries. The detailed software tools and libraries of this proposed system have been listed below:

i. Python 3.6: Basic environment for the application

ii. NLTK python library: Natural language processing library comprise of basic NLP methods

iii. CLTK python library: In order to inherit the stop word list for Hindi language

iv. Itertools python library: To successfully generate various sense combination in order to evaluate them for score.

v. Indicnlp python library: For the morphological processing of data to support stemming.

6. APPLICATIONS OF THE PROPOSED SYSTEM

The proposed algorithm is useful in systems, which includes analysis of a query to recommend related queries. In case user is not able to exactly describe the desired query, related results are suggested to user by this system. This system has many applications like in Pension Systems, where senior citizen can use words with which they are comfortable. Sentiment analysis of news headlines in order to predict the stock prices is another possible business application of the proposed algorithm. Positive and negative news have vast impact on the present trend for a stock. An automated system predicting the effect of a news on a stock price can use our proposed algorithm. The dissected sense can be used to various text summarization problems. Text summarization can use word sense disambiguation as one of their sub problems. Proposed algorithm can enhance the performance of text summarization algorithm and improve the efficiency of a search engine by understanding what the user is actually looking for after dissecting the input query.
7. **Future Work**

Proposed algorithm finds many combinations for sense tokens. It is not possible always to evaluate each of the combinations to get the correct sense as an output. Because even after numerous combinations that generate several senses, the correct or desired sense may not be derived as the output. Thus, we may incorporate some heuristic that can approximate our output to a near optimal result. Though, it is observed that such kind of queries does not occur too often in the natural language domain. Future work lies on identifying one or two heuristic methods based on genetic algorithm that may be used to provide a near optimal solution. Even deep learning based algorithm may be used in future to identify the correct sense of an ambiguous word in Hindi sentences. Hypernym, hyponym, gloss of each sense consist of various stop words and words with no available senses. Incorporating these words affect the performance of the algorithm. One possible solution to eliminate these words is to process them again by performing morphological analysis. But re-processing will increase the time complexity of the algorithm. As discussed earlier, this algorithm depends hugely on the semantic collection of words and their meanings provided by the WordNet. So the performance of the algorithm depends hugely on the WordNet. Dataset generation is a major step in the proposed algorithm. As it segregates only a portion of the data from WordNet. Due to which the database needs to be queried again and again. So an efficient execution of querying operation of WordNet can further enhance the performance of the proposed algorithm. The proposed system can be enhanced using machine learning algorithm or deep learning algorithms.

8. **Conclusion**

The proposed algorithm provides a word sense disambiguation solution for the Hindi language. The proposed method rightly, gives pair of words and their evaluated senses as an outcome of the system. Novel scoring method used with Lesk alogorithm gives improved performance in disambiguating the ambiguous words for the Hindi language. Hindi WordNet database designed by CFILT IIT Bombay is very useful in solving Hindi word sense disambiguation problem. To improve the performance of Lesk algorithm, we have used the scoring method which assigns scores to the sense of the tokens depending on the combinations of sense tokens. Word gloss, hypernym, hyponym and synonym are very useful for calculating the sense score. The time complexity of the proposed algorithm is $O(n^2)$ where $n$ is the number of tokens. Though algorithm imposes restriction on the length of the query sentence, it yields a good performance over constrained input data. Despite unavailability of large amount of ambiguous data involving polysemous words in Hindi language for testing, the proposed algorithm successfully dissects the sense for most of the test queries provided.

**Acknowledgment**

This research work is done at the Research Project Lab under Dept. of Computer Science and Engineering of National Institute of Technology (NIT), Durgapur. The authors would like to thank Dept. of Computer Science and Engineering, NIT, Durgapur, India for academically supporting this research work.

**References**


AUTHORS

Prafullit Tripathi is a Software Engineer in Qualcomm Company. He has completed his B.Tech-IT degree from National Institute of Technology, Durgapur, India. He has successfully completed his summer internship at Qualcomm Company. He has a good interest in Implementation of Natural Language Processing based Applications Disambiguation, Query-Response Model, like Word Sense Question Answering System, Sentiment Analysis, etc.

Prasenjit Mukherjee has 12 years of experience in academics and industry. He was a fulltime Ph.D. Research Scholar in Computer Science and Engineering in the area of Natural Language Processing from National Institute of Technology (NIT), Durgapur, India under the Visvesvaraya PhD Scheme from 2015 to 2019. Presently, He is working as a Data Scientist under Analytics and IT Department, RamanByte, Pune Institute of Business Management, Pune, Maharashtra, India.

Manik Hendre has 5 years of experience in academics and industry. He is a Ph.D. research scholar at the University of Pune. He is currently working as a Data Scientist in RamanByte Pune. His research areas include Biometrics Image Processing, Machine learning and Data Analytics.
Dr. Manish Godse has 25 years of experience in academics and industry. He holds Ph.D. from Indian Institute of Technology, Bombay (IITB). He is currently working as an Industry Professor and IT Director in the PIBM, Pune in the area of Artificial Intelligence and Analytics. His research areas of interest include automation, machine learning, natural language processing and business analytics. He has multiple research papers indexed at IEEE, ELSEVIER, INDERSCIENCE, etc.

Dr. Baisakhi Chakraborty received the PhD. degree in 2011 from National Institute of Technology, Durgapur, India in Computer Science and Engineering. Her research interest includes knowledge systems, knowledge engineering and management, database systems, data mining, natural language processing and software engineering. She has several research scholars under her guidance. She has more than 60 international publications. She has a decade of industrial and 14 years of academic experience.

http://journals.uob.edu.bh
Response of Reviewer Comments

Paper #1570637815 (‘Word Sense Disambiguation in Hindi Language Using Score Based Modified Lesk Algorithm’)

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Reviewer Comments</th>
<th>Response of Reviewer Comments</th>
</tr>
</thead>
</table>
| 1      | 1. The paper is presented in a very comprehensive way.  
2. Results are very interesting.  
3. Authors collected very interesting data and used python as a programming language and data mining techniques to data mining and processing.  
4. I would accept this paper as novel research work and recommend to accept this paper. | 1. Thank you to Reviewer 1 for his/her positive comments on our research paper.  
2. Results have been given clearly in this research paper and thank you to Reviewer 1 for his/her positive comments on Results.  
3. Python language has been used because today python is a most famous computer language in computer science. Hindi WordNet of IIT, Bombay, India has been used in this WSD because WordNet on regional languages is still in developing state. The IIT, Bombay has developed Hindi WordNet which is very useful for NLP researcher.  
4. Thank you to Reviewer 1 for recommendation to accept our paper. |
| review 2  | 1. The problem undertaken is quite interesting and important. Solution to the Word sense disambiguation (WSD) problem will open opportunities to Hindi savvy people to use relevant web applications.  
2. The authors have used Lesk algorithm improved by novel scoring method.  
3. The results are encouraging.  
4. Larger size of the data set could have strengthened the validation of the solution.  
5. Implementation of the Lesk algorithm in conjunction with improved scoring method makes a fair contribution towards solving the problem of WSD.  
6. It will open the opportunities in web based applications including query-response system, question-answer system and sentiment analysis.  
7. A promising solution to WSD has been proposed | 1. Thank you to Reviewer 2 for his/her positive review comment on our research work.  
2. The Lesk algorithm has been modified by using scoring method. Lesk is a most popular algorithm in word sense disambiguation. The Lesk algorithm has been used to implement a WSD system on Hindi language.  
3. Thank you to Reviewer 2 for his/her positive comment on results.  
4. Yes, larger size of the data set could have strengthened the validation of the solution. But getting larger size of dataset on regional language is difficult. We will improve our system accordingly which has been discussed in the future work section of the paper.  
5. Thank you to Reviewer 2 for his/her positive comment on implementation of the Lesk algorithm.  
6. Yes, WSD system can be used in web based applications like query-response system, question-
and reasonably validated against the HindiWordNet data base.

8. The results are important.

answer system and sentiment analysis. The proposed WSD system can be modified for multiple regional languages and that will be an added advantage for such applications.

7. The Hindi WordNet has been designed by the IIT, Bombay which is very useful for NLP research or NLP based system implementation. Thank you to Reviewer 2 for his/her positive comment.

8. Thank you to Reviewer 2 for his/her positive comment on results.

<table>
<thead>
<tr>
<th>Reviewer 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
</tr>
<tr>
<td>1. The architecture and the methodology of the proposed WSD system is well builded and clearly explained.</td>
</tr>
<tr>
<td>2. From a scientific point of view, the paper lacks of novelty and scientific innovation. Today, modern WSD solutions include machine learning techniques.</td>
</tr>
<tr>
<td>3. From a scientific point of view, the paper lacks of novelty and scientific innovation.</td>
</tr>
<tr>
<td>4. The paper is in fact a well and properly done study that involves a fairly modification of the most popular algorithm in this field.</td>
</tr>
</tbody>
</table>

1. Thank you to Reviewer 3 for his/her positive comment on the architecture and the methodology of our proposed WSD system.

2. As we all know, the accuracy of the machine learning algorithm highly depends on the available training data. NLP based system implementation on regional language is very difficult. Because the unavailability of regional language data. Computer supporting script, regional language dataset generation are the other major issues. We have implemented our WSD system using Hindi WordNet by IIT, Bombay. Lesk is a most popular and successful algorithm in WSD. The Lesk algorithm has been modified by scoring method. The Machine Learning or Deep Learning algorithm can be useful to get better result in Hindi Word Sense Disambiguation and we have mentioned it as our further enhancement in Future Work section.

3. The Lesk algorithm has been modified using the novel scoring method. The traditional Lesk algorithm is not sufficient to solve the Hindi Disambiguation problem. The scoring method has been used to disambiguate the Hindi words. We have got good results on Hindi sentences and Time complexity of proposed system has been given which is O(n^2). Word Sense Disambiguation on regional languages is an open research problem and it is less researched.

4. The paper has been modified again according to Reviewer 3 comments and Thank you to Reviewer 3 for his/her positive comments.

<table>
<thead>
<tr>
<th>Reviewer 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
</tbody>
</table>
| 1. The paper contains interesting algorithm and practical issue for the ease of using web application.| 1. Yes, WSD system can be used in web based applications like query-response system, question-
applications for unfamiliar users with English, which considered important issue. However, the language of the work must be written in good English language which was not, there are a lot of English semantic and syntax errors, example from the abstract: Being many citizens aren’t using English, the major challenges in to manage Ambiguous Hindi words, the inappropriate meaning always leads to chaos and increase support load. So I suggest the writers to revise the paper and resubmit.

2. The work is well analytical and procedural methods for addressing the problem of catching the semantic of the Hindi, still the algorithm must be tested by developing a testing experiments for well evaluation. from my point view further practical experiments are a must.

<table>
<thead>
<tr>
<th>Reviewer 5</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The article contains good information and The algorithm was applied to a set of characters that no other article used.</td>
<td>1. Thank you to Reviewer 5 for his/her positive comment on the algorithm of our proposed WSD system.</td>
</tr>
<tr>
<td>2. The article does not contain modernity, but it discusses a topic containing the application of the algorithm to the Indian characters.</td>
<td>2. This paper has been revised according to the Reviewer 5 comments. The proposed system has been discussed clearly with architecture and methodology. The Hindi WordNet has been used to implement this system that is also available in IIT, Bombay websites. A comparative analysis among similar type recent WSD systems with proposed WSD system has been done in Related Works section. The Time complexity has been calculated which is O (n²).</td>
</tr>
<tr>
<td>3. One of the most important notes: a)- The research needs to be reformulated b)- The large number of shapes in the article is not desirable c)- Most shapes can be combined and made into one table d)- The equation for the algorithm used must be written and the variables used in the algorithm defined</td>
<td>3. a) The paper has been revised according to the Reviewer 5 comment.</td>
</tr>
<tr>
<td>b) Unused Shape (Figure 2) has been removed according to the comment of Reviewer 5. Each shape and each table is containing different information and that are given in different steps of algorithm or in sections. For better clarity, each shape and each table are needed for large audiences. Figures and Tables have been given to understand the work of proposed WSD system properly.</td>
<td></td>
</tr>
<tr>
<td>c) Each shape and each table is containing different information and that are given in different steps or sections. The characteristics of proposed Hindi WSD system and characteristics Hindi words are clearly visible in these Figures tables.</td>
<td></td>
</tr>
</tbody>
</table>
| d) The equation of proposed algorithm has been given from Dataset Generation to Output subsection of answer system and sentiment analysis. The proposed WSD system can be modified for multiple regional languages and that will be added advantage for such applications. This paper has been revised and grammatical and language corrections have been done according to Reviewer 4 comment to eliminate semantic and syntax errors.
Algorithm Section (Section 3.1) in Architecture of Proposed System (Section 3). The each step of algorithm and its Pseudo code in Python has been clearly described. Today Python is a most popular computer language which is easy to understand for large audience. Variables are defined in pseudo code are also easily understandable for the audience.

<table>
<thead>
<tr>
<th>Managing Editor Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>