NATURAL LANGUAGE PROCESSING USING SOFT COMPUTING

Thesis submitted to the Delhi Technological University in partial fulfilment of the requirements for the award of the degree of

DOCTOR OF PHILOSOPHY

In

Computer Science and Engineering

By

Ms. Minni Jain

Under the supervision of

Prof. Rajni Jindal

Computer Science & Engineering Department

Delhi Technological University

Dr. Amita Jain

Computer Science & Engineering Department

Netaji Subhas University of Technology

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

DELHI TECHNOLOGICAL UNIVERSITY

NEW DELHI

2023

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

DELHI TECHNOLOGICAL UNIVERSITY

NEW DELHI

DECLARATION

I hereby declare that this thesis titled **"Natural Language Processing using Soft Computing"** submitted by me to the Department of Computer Science & Engineering of Delhi Technological University, New Delhi for the award of the degree of **Doctor of Philosophy (Ph.D.)** In **Computer Science and Engineering is** a bona fide work carried out by me under the supervision of Prof. Rajni Jindal (CSE, DTU) and Dr. Amita Jain (CSE, NSUT).

The matter embodied in the thesis has not been submitted to any other University or Institution for the award of any other degree or diploma.

Minni Jain

Minni Jain
2k17/PhD/CO/12

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

DELHI TECHNOLOGICAL UNIVERSITY

NEW DELHI

CERTIFICATE

This is to certify that this thesis titled **"Natural Language Processing using Soft Computing"** submitted by **Ms. Minni Jain** to the Department of Computer Science & Engineering of Delhi Technological University, New Delhi for the award of the degree of **Doctor of Philosophy (Ph.D.)** In **Computer Science and Engineering** is a bona fide work carried out by her under our supervision.

The matter embodied in the thesis has not been submitted to any other University or Institution for the award of any other degree or diploma.

Rajni Jindal

Prof. Rajni Jindal (Supervisor)
Computer Science Engineering Department
Delhi Technological University

[illegible] 22/9/23

Dr. Amita Jain (Co-Supervisor)
Computer Science Engineering Department
Netaji Subhas University of Technology,
East Campus

Acknowledgement

With a deep sense of gratitude, I wish to express my sincere thanks to my supervisor, Prof. Rajni Jindal for her immense help in planning and executing this thesis work in time. She consistently stood by me in all my difficult times helping me to do my research fruitfully. I would like to express my sincere gratitude to my co-supervisor Dr. Amita Jain, for her excellent guidance, inspiration and support throughout my research work. I am grateful to her for the time she has spent with me in discussions and giving valuable suggestions. Their enthusiasm, innovative suggestions, and deep insight into technical matters were instrumental in the conceptualization of this work. Working under their supervision was a great learning experience.

I owe deep sense of gratitude to the Prof. Vinod Kumar, HOD of the Department of Computer Science and Engineering, SRC, DRC members, faculty members, and research scholars, staff of the department for their valuable support and time to time help. Finally, I would like to express my heartfelt regards to my family.

There are exceptional mentors that I acknowledge due to their importance in my work. Since I can't name all of them, I owe my debt and extremely warm gratitude to all of whose honorable references I have referred to in my reference section while working on the thesis.

Minni Jain

Ms. Minni Jain
Roll No: 2K17/PHD/CO/12
Department of Computer Science & Engineering
Delhi Technological University, Delhi-110042

Abstract

Natural Language Processing (NLP) is a field of study that focuses on the interaction between computers and human language. It is an area related to computational linguistics, computer science and artificial intelligence. NLP has many applications including text summarization, sentiment analysis, text classification, keyword extraction, information retrieval etc.

This thesis proposes different models to solve various NLP problems such as text normalization of code-mixed social media text, word sense disambiguation, interval-valued fuzzy Hindi WordNet lexicon generation, domain-specific sentiment lexicon generations, text summarization and keyword extraction. The proposed models used fuzzy graphs, interval-valued fuzzy graphs, game theory and various word embeddings to solve the above-mentioned problems. A brief description of our contribution to NLP areas is discussed as follows:

- This thesis proposed a novel text correction framework for code-mixed Hindi-English social media text. The proposed method for the first time handles language identification along with non-word and real-word errors in code-mixed text.
- This study proposed the first research for word sense disambiguation that uses fuzzy WordNet and BERT embeddings to automatically assign the membership values to fuzzy graphs.
- This research work proposed a novel method to improvise the famous and widely used NLP lexicon i.e. Hindi WordNet. This work applied the concept of Interval-Valued Fuzzy graphs and proposed Interval- Valued Fuzzy Hindi WordNet (IVFHWN).
- This thesis proposed methods for domain-specific sentiment lexicons, especially for sentiment analysis. One method for the first time proposed *DoSLex* unsupervised, language-independent framework with contextual semantics based for low resource languages. Two methods are proposed for domain-specific lexicon generation using a random walk algorithm and label propagation on conceptnet graph.
- This thesis presents novel methods for text summarization and keyphrase extraction using game theory and various word embeddings such as m-bert, fastext, glove and word2vec.

The proposed research works in this thesis are compared with existing state-of-the-art methods of relevant areas. The study exhibits improved performance compared to the state-of-the-art methods. It is observed that soft computing approaches viz. fuzzy graphs, fuzzy graph centrality measures, interval-valued fuzzy graphs, word embeddings, and embeddings-based game theory are able to solve complex, uncertain and ambiguous NLP issues.

<u>**CONTENTS**</u>

Page No.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

NLP	Natural Language Processing
FGCM	Fuzzy Graph Connectivity Measures
HWN	Hindi WordNet
WSD	Word Sense Disambiguation
LSTM	Long Short-Term Memory
SVM	Support Vector Machine
IVFHWN	Interval-Valued Fuzzy Hindi WordNet
AMT	Amazon Mechanical Turk
DFS	Depth First Search
OOV	Out-of-vocabulary
WSD	Word Sense Disambiguation
LSI	Lexical Semantic Identification
MT	Machine Translation
IR	Information Retrieval
IIT-B	Indian Institute of Bombay
FHWN	Fuzzy Hindi WordNet
OMCS	Open Mind Common Sense
POS	Part of Speech
MSE	Mean Square Error
VP	Verb Phrase
NP	Noun Phrase
EV	Expert Value
SA	Sentiment Analysis

Chapter 1

Introduction

Natural language processing (NLP) is a tract of Artificial Intelligence and Linguistics, devoted to make computers understand/generate the statements or words of human languages. There are many application areas of NLP including Search Engines, Sentiment Analysis, Text Summarization, Word Sense Disambiguation, Keyword Extraction and many more.

The overall framework of NLP includes input data sources, preprocessing techniques, knowledge bases/lexicons and NLP algorithms (shown in Fig. 1.1). Input data sources can further be classified into structured and unstructured data sources. Structured data is factual and highly organised for instance news articles, research papers etc. while unstructured data is informal and do not follow any particular format for instance chat messages, blogs, social media posts etc. The next major component is pre-processing techniques. Preprocessing techniques completely depend on what kind of input data and what type of NLP problem is going to solve. Some standard pre-processing techniques are tokenization, stop word removal, and part-of-speech tagging. Unstructured data needs more pre-processing due to the presence of noisy elements such as slang terms, abbreviations, Unnecessary excess use of punctuation and symbols and many more. Improper pre-processing of input text directly affects the performance of NLP algorithm so it is a crucial step. In this thesis one chapter is dedicated on text normalization of social media Hindi-English code-mixed text and its impact on performance.

Knowledge bases/Lexicons are the backbone of any language. They are different from dictionaries as they comprise words along with their definition, usage, origin and their relation with other words of the lexicon. In literature, there are many lexicons available for instance WordNet, ConceptNet and some are application based such as SentiWordNet, SenticNet etc. But these existing lexicons are not close to real-life scenarios. This thesis discussed the proposed improvised lexicons that are closer to real life and can handle real-life uncertainties. The last major component of NLP is NLP techniques used for problem solving. NLP techniques can be classified in to machine learning and lexicon-based techniques. This thesis used both type of techniques to handle different problems of NLP.

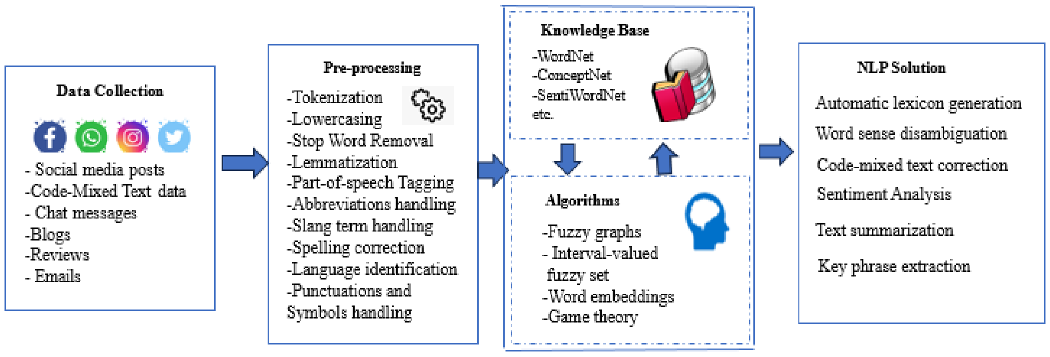


Fig 1.1. Framework of Natural Language Processing

1.1 Proposed Approaches

This thesis introduces various novel methods to address the several problems of natural language processing. These problems primarily focus on text correction for Hindi-English code-mixed social media text, word sense disambiguation, automatic generation of interval-valued fuzzy Hindi WordNet, domain-specific sentiment lexicon generation, text summarization and keyphrase extraction. To handle above-mentioned problems, this thesis incorporates various soft computing methods like fuzzy graphs, fuzzy graph connectivity measures, interval-valued fuzzy graphs and word embeddings.

The proposed approaches are precisely mentioned below: -

i. This thesis introduced a novel method for code-mixed Hindi-English social media text normalization a method proposed that comprises language identification, detection and correction of non-word (Out of Vocabulary) errors as well as real-word errors occurring simultaneously. Interaction via social media involves frequent code-mixed text, spelling errors and noisy elements, which creates a bottleneck in the performance of natural language processing applications. To perform text correction, we utilized WordNet, fuzzy graph connectivity measures and Word2vec embedding. Several experiments are performed on different social media datasets taken from Instagram, Twitter, YouTube comments, Blogs and WhatsApp. The experimental results demonstrate that the proposed system corrects out-of-vocabulary words as well as real-word errors with a maximum recall of 0.90 and 0.67, respectively for Dev_Hindi and 0.87 and 0.66, respectively for Rom_Hindi.

ii. This thesis presents a novel method proposed for WSD of text using fuzzy WordNet. Fuzzy WordNet has been shown as a better resource for representing relations between concepts. Membership values of fuzzy relations in fuzzy Hindi WordNet were taken from experts' opinions, which may be ambiguous. In the proposed algorithm, we assign the membership values to fuzzy relationships using BERT embedding. We then perform WSD using fuzzy graph centrality measure. The method is compared with the existing approaches, and the results are encouraging.

iii. This work proposed a method to improvise the famous and widely used NLP lexicon i.e. Hindi WordNet. A computational lexicon is the backbone of any language processing system. It helps computers to understand the language complexity as a human does by inculcating words and their semantic associations. Manually constructed famous Hindi WordNet (HWN) consists of various classical semantic relations (crisp relations). To handle uncertainty and represent Hindi WordNet more semantically, Type- 1 fuzzy graphs are applied to relations of Hindi WordNet. But uncertainty in the crisp membership degree is not considered in Type 1 fuzzy set (T1FS). Also collecting billions (5,55,69,51,753 relations in HWN) of membership values from experts (humans) is not feasible. This work applied the concept of Interval-Valued Fuzzy graphs and proposed Interval- Valued Fuzzy Hindi WordNet (IVFHWN). IVFHWN automatically identifies interval- valued fuzzy relations between words and their degree of membership using word embeddings and lexico-syntactic patterns. The experimental results for the word sense disambiguation problem show better outcomes when IVFHWN is being used in place of Type 1 fuzzy Hindi WordNet and classical Hindi WordNet.

iv. This thesis also introduces methods for domain-specific sentiment lexicons, especially for sentiment analysis. One method proposed *DoSLex*, where, all the words are represented in a circle where the centre is the domain, and the *x* & *y* axis denote the strength and sentiment orientation, respectively. In the circle, the radius is the contextual similarity between the domain and term calculated using MuRIL embedding, and the angle is the prior sentiment score taken from various knowledge bases., Another two methods are proposed using random walk algorithm, label propagation on concept net graph.

v. This thesis presents methods using game theory for natural language processing applications. Proposed methods applied game theory with embeddings in text summarization and keyphrase extraction.

In the next section, this chapter describes the basic concepts of various natural language processing and soft computing like Hindi WordNet, word embeddings, game theory, fuzzy set theory, fuzzy graphs, fuzzy graph connectivity measures and interval-valued fuzzy graphs.

1.2 Preliminaries

This section briefly introduces some key concepts and terminologies with the mathematical terms required to understand the approach discussed in forthcoming chapters.

1.2.1 Hindi WordNet (HWN)

Among the various Indian language WordNets, the Hindi WN is the first WordNet that came into existence from 2000 onwards. HWN is inspired by English WN, and it is created manually using lexical knowledge acquired from various dictionaries. It comprises nouns, adjectives, verbs and adverbs structured into synonym sets, each demonstrating one lexical concept. HWN is used in various natural language processing applications like Word Sense Disambiguation [12], Sentiment Analysis and Opinion Mining [13], Information Retrieval [6] and many more. WordNet is a word sense network that can be represented as a graph also. A word sense node in this network is a synset. Each synset is linked with other synsets through lexical and semantic relations like hyponymy, hypernymy, troponymy, entailment, antonymy etc. Hindi WordNet consists of 17 relations, including same part-of-speech and cross part-of-speech relations.

1.2.2 Word Embeddings or Word Vectors

Word Embedding or Word Vector is a recent development in computational linguistics. Many deep learning-based natural language processing applications use it as a fundamental resource. A word embedding is a vector representation of a word with the property that the embeddings of comparable words in an N-dimensional vector space are close to each other. Word2Vec is a common technique for learning word embeddings that employ shallow neural networks. [14]. Cosine similarity is a metric to measure the distance between two words. Word2Vec can be used to find the belongingness between two words in a fuzzy graph.

1.2.3 Evolutionary Game Theory

Evolutionary Game Theory was first brought to light by Smith and Price in 1973. Evolutionary game theory induces an *inductive learning* process wherein a set of entities play games with other entities (called neighbors) repeatedly. The players update their understanding of the shape of the game at each iteration. Further strategies are developed according to what has proven to be fruitful previously. The strategy space of a player *i* can be defined as a mixed strategy profile $x_i = (x_1, \dots, x_m)$, where *m* is the count of pure strategies and x_h denotes the probability of player *i* choosing the *h*th pure strategy. In every iteration, a player updates its strategy space in accordance with the payoffs obtained as a result of the games. It then assigns greater probability to the strategy which has gained higher payoff and the process is continued until a point of equilibrium is achieved – Nash Equilibrium [21, 23].

1.2.4 Fuzzy Graphs and Fuzzy Graph Connectivity Measures (FGCM)

The notion behind fuzzy logic was that "everything is a matter of degree," and Zadeh in 1965 explained it using a mathematical framework [1]. A fuzzy set, then, maps each element to the value from 0 to 1 so that it is a matter of degree rather than rejection or confirmation (0,1). Words are included because there need not only be a binary relationship between two or more words; there also has to be a degree relationship. Therefore, fuzzy relations can be employed to infer a relationship between two word pairs. "A symmetric binary fuzzy relation on a fuzzy subset is a fuzzy graph. A Fuzzy Graph defined as $G = (\sigma,\mu)$ is a pair of functions $\sigma: S \rightarrow [0,1]$ and $\mu: S\times S \rightarrow [0,1]$, where for all x, y in S we have $\mu(x,y) \leq \sigma(x) \cap \sigma(y)$. Any relation $R \subseteq S\times S$ on a set S can be represented as a graph with node set S and arc set R [2,3,4]. Similarly, any fuzzy relation $\mu: S\times S \rightarrow [0,1]$ can be regarded as defining a fuzzy graph, where the arc $(x,y) \in S\times S$ has weight $\mu(x,y)\in[0,1]$" .

To select the most significant nodes and consequently the most pertinent phrases, our method uses fuzzy centrality measures to identify and compare the importance of nodes that are relative to one another.

Fuzzy Graph connectivity measures (FGCM) were proposed by Jain and Lobiyal [5]. It has been heavily used in wide applications of NLP [6,7]. FGCM help in finding the vitality of a particular node. In a group of nodes, FGCM can determine importance of every node in the given set of nodes. There are several measures which helps in determining the importance of node in a graph, such as fuzzy centrality, fuzzy HITS, fuzzy key player problem (KPP), fuzzy PageRank etc. The connectivity measures used in this thesis are as follows:

i. Fuzzy Degree Centrality: It takes into account a vertex's degree, which is determined by the number of connections that vertex has to other vertices. The degree for a vertex v (deg_v(v)) in a fuzzy graph is determined by the equation:

$$deg_f(v) = \sum_{u\neq v} \mu_{uv} \tag{1.1}$$

where (u, v) belongs to the set of edges E

Fuzzy degree centrality ($C_f(v)$) is a measure of a vertex's degree that is standardised by the highest degree as

$$C_f(v) = \frac{deg_f(v)}{|v|-1} \tag{1.2}$$

ii. Fuzzy Eigenvector Centrality: According to this centrality, connections to nodes with high scores "add more to the score of the node in question than similar connections to nodes with low scores." It comes in two flavours: Fuzzy PageRank and Fuzzy HITS [8].

For vertex (v_a) fuzzy PageRank is calculated in accordance with

$$v_a = \frac{(1-d)}{|V|} + d \sum_{(v_a, v_b)\in E} \frac{\mu_{ba}}{\sum_{(v_b, v_c)\in E} \mu_{bc}} (v_b) \tag{1.3}$$

where E is the set of edges, d is the damping factor, and d ∈ [0, 1], and μ_{ba} is the weight of the edge joining(v_a) and (v_b). In this post, we adopted the d = 0.85 value that Luca and Carlos suggested (2006). Every vertex's initial fuzzy PageRank score is given a score of 1. In contrast to fuzzy PageRank, the second form, fuzzy HITS, distinguishes between hubs and authorities.

Hubs $h_f(v)$ and authorities $a_f(v)$ have been proposed by Jain and Lobiyal for fuzzy graphs [5].

$$h_f(v) = \sum_{(u,v)\in E} \mu_{uv} a_f(u) \qquad (1.4)$$

$$a_f(v) = \sum_{(u,v)\in E} \mu_{uv} h_f(u) \qquad (1.5)$$

where μ_{uv} is the strength of the edge connecting vertices u and v. By combining the authority and hub values, we use it as a single measure HITS in our methodology.

iii. Fuzzy Closeness Centrality: It is based on the idea that a vertex's relative proximity to all the other vertices determines how essential it is. For a fuzzy graph, fuzzy closeness centrality (u_n) is defined as

$$\mathrm{u_n} = \frac{\sum_{u_0\in V, u_0\neq u_n} \frac{1}{min_{all\ paths\ u_0\ to\ u_n}[\delta(u_0,u_n)]}}{|v|-1} \qquad (1.6)$$

$\delta(u_0, u_n)$ is the length of path P: $(u_0, u_{1,\dots,}u_n)$ that exists between two vertices (u_0, u_n) and is computed as if a node is disconnected, its closeness centrality is given by $\frac{1}{|V|}$

$$\delta(u_0, u_n) = \sum_{i=1}^{n} \frac{1}{\mu_{(i-1)(i)}} \qquad (1.7)$$

iv. Fuzzy Betweenness Centrality: The concept of shortest paths is the foundation for this centrality. It is calculated for a vertex *v* as

$$fuzzy\ betweenness_f(v) = \sum_{s,t\ \in V, s\neq v\neq t} \frac{|d_v(x,y)|}{|d(x,y)|} \qquad (1.8)$$

The number of paths that pass via the vertex v is represented as $|d_v(x, y)|$ where $|d_v(x, y)|$ is the number of shortest paths from vertex *x* to vertex *y*.

Fuzzy $betweennesscent_f(v)$ is a fraction of fuzzy betweenness (v) to the largest pair of vertices that don't have *v* for a vertex in a fuzzy graph.

$$fuzzy\ betweennesscent_f(v) = \frac{betweenness_f(v)}{(|V|-1)(|V|-2)} \qquad (1.9)$$

1.2.5. Interval-Valued Fuzzy Set

An Interval-Valued Fuzzy set, for an element x in the universe of discourse *X*, can be defined as [9],

$$\tilde{A} = \{(x, [\mu_{\tilde{A}_L}, \mu_{\tilde{A}_U}]) | x \in X\} \qquad (1.10)$$

Where, $[\mu_{\tilde{A}_L}, \mu_{\tilde{A}_U}]$ is an interval between [0,1]. For all *x*, are the membership interval iff:

$$0 \le \mu_{\tilde{A}_L} \le \mu_{\tilde{A}_U} \le 1$$

Fig. 1.2. Illustrate the membership value at *x'* of the Interval-Valued Fuzzy Set Ã; thereby, the membership value of *x'* is the interval of $[(\mu_{\tilde{A}_L}, \mu_{\tilde{A}_U})]$

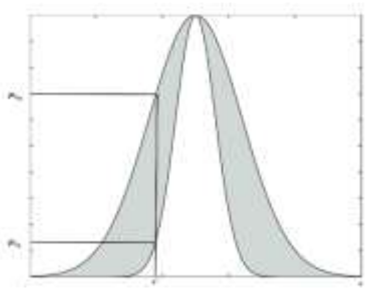


Fig.1.2. Interval-Valued Fuzzy Set Ã

1.2.6.Interval-Valued Fuzzy Graph

An Interval- Valued fuzzy graph *G* = (V, E)* is a pair *G** = (Ã, B̃) where Ã = $[\mu_{\tilde{A}_L}, \mu_{\tilde{A}_U}]$ is an Interval- Valued Fuzzy set on *V* with $0 \le \mu_{\tilde{A}_L}, \le \mu_{\tilde{A}_U} \le 1$and B̃ = $[\mu_{\tilde{B}_L}, \mu_{\tilde{B}_U}]$ is an Interval- Valued fuzzy relation *E* with $0 \le \mu_{\tilde{B}_L}, \le \mu_{\tilde{B}_U} \le 1$. [10,11]

Example: Consider a graph *G* = (V, E)* such that *V= {x,y,z}* and *E = {xy, yz, zx}*. Let Ã = $[\mu_{\tilde{A}_L}, \mu_{\tilde{A}_U}]$ be an Interval- Valued Fuzzy set on *V (vertex)* and Let B̃= $[\mu_{\tilde{B}_L}, \mu_{\tilde{B}_U}]$ be an Interval- Valued Fuzzy relation *E (edges).* Matrices for *V* and *E* are:

$$\begin{pmatrix} & x & y & z \\ [\mu_{A_L}, \mu_{A_U}] & [0.3,0.4] & [0.5, 0.7] & [0.4,0.6] \end{pmatrix}$$

$$\begin{pmatrix} & xy & yz & zx \\ [\mu_{R_L}, \mu_{R_U}] & [0.8,0.9] & [0.4, 0.7] & [0.3,0.4] \end{pmatrix}$$

1.3 Motivation

There are billions of text data being generated every day. This large volume of structured and unstructured text data can provide useful insights. This enormous text can be handled by NLP techniques. Bottleneck behind the limited performance of NLP techniques is ambiguity and uncertainty present in natural language text. Soft computing algorithm with NLP techniques can better deal with text ambiguity and uncertainty to make it more close to real-life scenarios. Thus it motivated us to take up the area of NLP using Soft Computing, where we constructed improved knowledge bases and applied to NLP applications

1.4 Research Gaps

- Lexicons are backbone of any language. WordNet available for most commonly used languages is based on crisp relationship between words. But in many real-life scenarios expressed in natural language, relations between words as a matter of degree.
- In existing work based on fuzzy knowledge base, the membership values for the relations are assigned by linguistics experts. No work is available for construction of automatically generation of fuzzy relations in WordNet.
- WordNet is a general purpose lexicon. It cannot be used for specific domain for instance technical and medical efficiently.
- Today, large volume of social media data is Code-Mixed text which need special prepressing techniques different from mono-lingual text.

1.5 Problem Definition

Natural Language processing is growing area and has various applications. NLP techniques widely use lexicons for solutions but these lexicons need improvements to make it more close to real-life scenarios. There is enormous text data available from social media having challenges such as mixing of languages (code-mixed text), intentional/unintentional spelling errors etc.. With the use of soft computing techniques various NLP applications such as word sense disambiguation, sentiment analysis, text correction, text summarization, keyphrase extraction, lexicon generations etc. performance can be improvised.

1.6 Objectives of the Research Work

The main research objectives of the work undertaken are:

Research Objective I- Identification of suitable soft computing technique for lexicon generation.
Research Objective II- Automatic generation of lexicon using soft computing techniques.
Research Objective III- Development of semantic methods for assigning membership values (for fuzzy logic systems) to the relations between words/ concepts.
Research Objective IV- Comparing performance of existing lexicons with proposed automatically generated WordNet for WSD. Incorporating common-sense in lexicon.
Research Objective V- Applying the automatic lexicon generation method for a sentiment analysis.
Research Objective VI- Generation of Domain specific Lexicon.

1.7 Organization of the Thesis

Finally, the organisation of the thesis is configured in the eight chapters as follows:

- **Chapter 1: Introduction**

This chapter presents the introduction and fundamental concepts of the research area followed by the problem statement. Further, this chapter encompasses the outline of thesis with a summary of the chapter at the end.

- **Chapter 2: Literature Survey**

The chapter explicates a fleet study of existing work related to the Research area that comprises Work related to Text normalization, word sense disambiguation, Lexicon generation, domain-specific lexicon generation and text summarization. Thereafter, the research gaps are identified and listed based on existing studies.

- **Chapter 3: Code-Mixed Hindi English Text Correction**

This chapter presents the proposed work for Code-mixed Text normalization. It covers basic knowledge of code-mixed language & its complexities, and algorithms to handle various types of errors in code-mixed text. The chapter also describes experimental details and results of different types of social media datasets.

Publication:

1. Minni Jain, Rajni Jindal, Amita Jain. "Code-Mixed Hi-En Text Correction using Fuzzy Graph and Word Embedding" Expert Systems Journal, Wiley, SCIE. IF- 3.3 DOI: https://doi.org/10.1111/exsy.13328

- **Chapter 4: Word Sense Disambiguation using Fuzzy Centrality Measures and BERT Embeddings**

This chapter illustrate lexical semantic analysis also called word sense disambiguation (WSD) with a new algorithm for handling WSD in Text. The methodology used various Fuzzy graph connectivity measures and BERT embeddings for WSD. It also presents the comparison of proposed methodology with state-of-art.

Publication:

1. Minni Jain, Rajni Jindal, Amita Jain "Lexical Semantics Identification using Fuzzy Centrality Measures and BERT Embedding" National Academy Science Letters, Springer, SCIE, IF- 1.1 DOI: https://doi.org/10.1007/s40009-023-01310-2

- **Chapter 5: Automatic Construction of Interval-valued Fuzzy Hindi WordNet**

This chapter elucidates an Interval-valued Fuzzy Hindi WordNet (a Lexicon) for various applications of Natural language processing. It presents the details of interval-valued fuzzy relations between words with their identification methods. The comparison of results with other studies demonstrates the effectiveness of the proposed Lexicon.

Publication:

1. Minni Jain, Rajni Jindal, Amita Jain "Automatic Construction of Interval-Valued Fuzzy Hindi WordNet using Lexico-Syntactic Patterns and Word Embeddings" *ACM* Transactions. (Major Revision Submitted)

- **Chapter 6: Domain Specific Lexicons for Sentiment Analysis**

This chapter presents the proposed methodologies for domain specific lexicons for sentiment analysis. It also explains the importance of domain specific lexicon over generalize existing lexicons through various experimentations.

Publications:

1. Minni Jain, Rajni Jindal, Amita Jain "DoSLex: Domain-Specific Lexicon for Unsupervised Sentiment Analysis of Low–Resource Languages" Language Resources and Evaluation, Springer. (Communicated)
2. Minni Jain, Rajni Jindal, Amita Jain "Building Domain-Specific Sentiment Lexicon using Random walk-based model on Common-sense Semantic Network" 6th International Conference on Innovative Computing And Communication (ICICC 2023), Delhi (Presented)
3. Minni Jain, Rajni Jindal, Amita Jain "Building Domain-Specific Sentiment Lexicon using Label Propagation" SCIS&ISIS 2018, Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS), Toyama,

Japan, December 5-8, 2018. IEEE 2018, ISBN 978-1-5386-2633-7, Japan. (Presented)

- **Chapter 7: Game Theory with Word Embeddings for Natural Language Processing**

This chapter presents the Game-Theoretic approaches for two different NLP applications that are Text Summarization and Keyphrase extraction.
Publications:

1. Minni Jain, Rajni Jindal, Amita Jain "Extractive Text Summarization using Game Theory with Word embeddings", Multimedia Tools & Applications, Springer, SCI. (Communicated)
2. Minni Jain, Rajni Jindal, Amita Jain "Keyphrase extraction using fuzzy based game theory", 4th International Conference on Data Analytics & Management (ICDAM-2023), Organized By: London Metropolitan University, London, UK (Venue Partner) in association with Karkonosze University of Applied Sciences, Jelenia Gora, Poland, Europe & Politécnico de Portalegre, Portugal, Europe &BPIT, GGSIPU, Delhi Date: 23rd - 24th June, 2023, London (Presented).

- **Chapter 8: Conclusion & Future Work**

The last chapter presents the conclusion in conjunction with the future scope of the present research work. It deliberates the importance of the proposed soft computing-based algorithms for lexicon generation and different natural language processing applications.

Chapter 2

Literature Survey

The enormous growth of social media platforms encouraged researchers to analyse social media texts efficiently. Various popular social media web platforms provide access to ample amounts of real-time data but thus data is noisy and unstructured. Analysis of this unstructured natural text is an important task. business. This chapter comprised various NLP research for text correction, word sense disambiguation, lexicon generation, sentiment analysis and text summarization. In this chapter, various research gaps and limitations have also been identified.

2.1 Text Correction for Hindi-English Code-Mixed Language

This section discusses the literature work done in the area of Hindi-English code mixed text, spelling correction on Devanagari Hindi and fuzzy graph connectivity measures.

For Hindi spelling error correction in literature, very limited work has been done. In Hindi, a method for spelling error correction and detection for non-word error has been proposed by Jain and Jain [15]. A system which detects and corrects document-specific error in Indic OCR using N-gram model has been proposed [16]. Gupta and Bag have also proposed a system for multilingual OCR based on polygonal approximation of words and the proposed system have tested on Hindi, Marathi, Punjabi and Bangla [17]. Another OCR for Hindi language based on long short-term memory (LSTM) is proposed for error detection and correction. The method has achieved the F-score around 92.4%. A supervised Support Vector Machine (SVM) based approach is proposed for text classifier [18]. This supervised approach has achieved high accuracy. There is an N-gram based approach for detection and correction of error in Hindi language [19]. Another N-gram based method has been proposed by Singh et al. (2019) for real-word error detection. They have used a file of 2000 Hindi words. They achieved precision value around 0.70 – 0.75, recall around 0.80 – 0.85 and F-score around 0.70 – 0.80. A deep learning-based model is also proposed [20]. UTTAM (Jain and Jain 2018) proposed a system for spelling error detection and correction using supervised approach in Hindi language. They have proposed to generate a candidate list followed by applying the Viterbi algorithm to find the most suitable sequence of words. They have achieved high accuracy as compared to previously available approaches [21]. A word prediction method was also introduced by authors. Currently, UTTAM is the only system which works for both real-word and OOV errors. The proposed system also works for both real-word and OOV errors and is the first approach using unsupervised learning which doesn't require any training data. In this thesis, our proposed work is covering both OOV as well as real-word errors for social media.

In literature, it is mentioned that the concept of fuzzy logic can deal with the uncertainties of the real world [1]. The proposed approach uses the fuzzy graph connectivity measure for the identification of the most important and contextually appropriate node. An unsupervised graph-based approach has been introduced [22], they have used their framework to find the most intended sense by recognizing the most important node in a graph. For word sense disambiguation, the idea of fuzzy graph connection metrics has been put up. Another method fuzzes the WordNet graph before using the connection measurements to determine the meaning of words [23]. Additionally, these techniques have been applied in numerous NLP applications. Fuzzy centrality measures have been used to do query expansion for the Hindi language. These techniques have also been employed to extract keywords from students' brief evaluation responses, and the results have been encouraging [6]. In all of the aforementioned works, the membership value for the fuzzy set was determined by linguistic specialists; however, in the suggested method, the membership value is determined automatically.

2.2 Word Sense Disambiguation

In recent years, WSD has garnered a lot of interest from researchers. Supervised, semi-supervised as well as unsupervised methods have been explored in the past [24]. Supervised machine learning techniques, along with topic modelling, have been used for disambiguation [25]. One of the semi-supervised methods used label propagation that is, it provides labels to the unlabelled sentences with the help of the similarity between them and the labelled sentences [26]. Certain unsupervised techniques use knowledge bases and are hence called knowledge-based methods [27, 28, 29].

Graph-based methods have recently gained a lot of popularity. They have been able to provide superior results without using any training data. An extension is the usage of fuzzy graphs [7, 23]. It has been observed that they provide better results than traditional supervised WSD systems. In a work experts have been asked to provide the membership value between two nodes [5]. Word embeddings have also been used along with supervised methods [7].

2.3 Lexicon Generation

Lexicons play an essential role in natural language understanding. Applying fuzzy set theory on widely used lexicon can improve the performance of lexicon-based NLP applications. In the early nineties, the author offered space as a network of fuzzy relationships called fuzzy concept networks by introducing four types of fuzzy relationships between the concept that are a positive association, negative association, specialisation and generalisation [30].

Further, this study of Concept networks was used [31] to enhance information retrieval performance. Since constructing the Fuzzy concept network manually was a tedious task, a method to automatically construct the fuzzy concept network was developed [32]. In 2015 Jain et al. got inspired by the idea of fuzzy concept network [5]. They proposed a fuzzy Hindi WordNet where authors assigned fuzzy membership value to the seventeen relations available in crisp Hindi WordNet. They assigned membership value with the help of various experts. The authors also show the performance of proposed Hindi fuzzy centrality measures and Hindi WordNet by applying on word sense disambiguation using various graph centrality measures. In Jain et al., authors applied the concept of Fuzzy Hindi WordNet on Hindi query expansion which also disambiguates the ambiguous query. Results of their work show that Fuzzy WordNet outperforms the crisp WordNet in the field of query expansion, too [6].

WordNet is available for several languages, including English, Hindi, Swedish, Portuguese etc. Many authors have applied the concept of fuzzifying the components of WordNet (nodes, edges, synsets). For instance, in 2009, authors presented an algorithm to construct Swedish fuzzy synsets [33] and in 2011. The authors proposed Portuguese fuzzy synsets [34]. Similarly, in 2021, Yousef et al. suggested interval probabilistic fuzzy WordNet for the English language [35]. They again fuzzified synsets of WordNet-based on the probabilistic fuzzy concept. Work by Sonakshi et al. in 2018 provides fixed weight to relations in WordNet for the English language [23]. Assigning a crisp membership degree does not account for uncertainty associated with methods by which membership values are computed. However, the proposed work is the first work that uses Interval-Valued Fuzzy (generalise form of type -2 fuzzy logic) to create IVF WordNet for the Hindi language. IVFHWN automatically assigns relations and interval membership between words using lexico-syntactic patterns and Hindi word embeddings.

2.3.1 Domain-specific lexicon Generation

This section discusses the existing literature for lexicons available for both Generic or domain-specific sentiment analysis developed for the Hindi language. In Literature, there are different methods for lexicon generation such as manual construction, dictionary-based or corpus- based approaches. The manual construction method, polarities to words are assigned by linguistic experts. This method is generally less adopted by researchers because it is time-consuming and it's tough to get sufficient experts. On the contrary, dictionary-based methods, uses a small set of seed words comprising sentiment words with their polarity. This list is then extended iteratively by different methods. Corpus-based approaches, extract syntactic patterns from a large corpus to produce sentiment words [36]. To get the polarity of words, there is a requirement of lexicon and Hindi language has such resources for instance Hindi Senti-WordNet (HSWN) [37]. HSWN was created manually by IIT Bombay and this resource is developed using English SentiWordNet and linking of English-Hindi WordNet. Hindi SentiWordNet consists of synsets that are labelled with positive, negative and neutral scores attached with their senses. HindiSentiWordNet is used as a base for many other developments in Hindi sentiment analysis. Next authors automatically and semi-automatically generated a sentiment lexicon using SentiWordNet [38]. This lexicon was created for Hindi, Bangla and Telugu languages. A lexicon for adjectives and adverbs is developed by Bakliwal et al. [39] using Hindi WordNet and they named it as HSL (Hindi subjectivity lexicon). Authors used Breadth-first search (BFS) and Hindi WordNet to build HSL. A method to improve Hindi Senti WordNet by increasing its coverage by adding more subjective words was proposed by Mittal et. al. [40]. This method also handled negation during the sentiment classification task. The experimental dataset is taken from domain movies only. Next, Arora et al. constructed a sentiment lexicon using the graph theory approach. Graph is created using WordNet, where nodes are the words and edges are the relations, specially synonymy and antonymy between word taken from WordNet [41]. This work utilized a pre- annotated small seed list in the beginning.Next Sharma et al. [42] proposed a bootstrap approach to create Hindi SentiLexicon that covers Nouns, Adjectives, verb and adverbs. They developed multi-module rule based sentiment analysis system. To create Hindi SentiLexicon, one positive and one negative seed words are considered. For every key word from the list, all the sense is extracted and if any of these words belong to opposite polarity then the sense is discarded else added in the lexicon with polarity. A method by Jha et al. [43] implemented both lexicon-based classification and machine learning techniques on Hindi movie reviews dataset. Authors extracted only adjectives and used them as polarity words for classification. They also used negation handling to improve the accuracy of methodology. In [44] Mishra et al. built a CSPL (context-specific polarity lexicon) resource for hotel and movie domain in Hindi. They also compared it with standard Hindi SentiWordNet. Authors also suggested some improvisation, including antonyms, larger datasets, and more. Modi et al. proposed a part-of-speech tagging method using rule-based approach. For known words, authors used corpus matching and for unknown words, grammar rules were used. This work is for the Hindi language [45]. Jha et al. [46] developed a sentiment-aware dictionary to handle the problem of multiple-domain sentiment classification. Here proposed dictionary consists of nouns, adverbs, adjectives and verbs built to classify amazon product reviews. Hussaini et al. [47] implemented a score-based sentiment analysis technique with word sense disambiguation handling. They considered book reviews for experimentation. An algorithm for lexicon-based SA is proposed by Yakshi Sharma et al. [48] and applied to Hindi tweets of “JAI HIND” and “World Cup 2015” topics. A subjectivity lexicon named as HSAS (Hindi subjectivity analysis system) was proposed by Jha et al. [49]. This lexicon was built using two methods first was by using an English subjectivity lexicon that is opinion –Finder and translating it and the second method used a method for seed list expansion. This second method used PMI (Pointwise mutual information) score and thus expanded 60 seed words to around 4500 polarity words. The PMI [50] score gives the confidence score of how much is a particular word is associated with a sentiment. Garg et al. [51] utilized lexicon-based method to perform sentiment analysis of the tweets related to the topic “Mann ki Baat by prime minister Mr. Narendra Modi”. Authors used a generic Hindi SentiWordNet lexicon in this approach. Rich Sharma et al. [52] created a sentiment-oriented system for movie reviews in the Hindi language, they also handled negation in text and achieved 65% accuracy. Authors also compared machine learning and domain-specific-lexicon approach for sentiment analysis [53]. A hate speech lexicon is created by Sayani et al. [54].

2.4 Text Summarization and Keyphrase Extraction

Text summarization has been an active area of interest in the research community for a very long time. Most of the studies in this area focus on the extractive summarization. Over the years, the foundation of natural language has shifted towards mathematical models. Applications of statistical techniques include language modelling for automatic speech recognition using smoothed n-grams, part-of-speech tagging using hidden markov models, syntactic parsing using probabilistic grammars and word sense disambiguation. The recent robust graph-based methods have proven to be beneficial in areas of natural language processing like keyword extraction [55], text summarization using lexical centrality and word clustering [56]. The search for an optimal solution to the summarization problem has led to an extensive exploration of both extractive and abstractive methods.

Early methods of extractive summarization were based on simple features like the frequency of the words of the sentence, the semantic nature of the sentences in a document and the key phrases indicating the importance of those sentences. Approaches that were explored involved the representation of the documents and their positions, lexical chains [57, 58] and cue and stigma words.

The machine learning approach for text summarization involved classifying the sentences of the text as summary and non-summary sentences. The model was trained using a reference dataset, learning the classification probabilities at the time of training. This approach includes both unsupervised clustering methods [59] as well as supervised learning methods such as classification-based techniques [60]. The supervised learning-based approaches [61] require labelled summaries at the training phase whereas the unsupervised systems [62,63] mostly rank sentences using some heuristics and do not require manually labelled summaries for training.

A common method known as term frequency-inverse document frequency (tf-idf) [64] reflects on how important a word in a document is using the measure inverse document frequency or *idf* (formulated as log(N/n_i), where N denotes the count of the documents in a collection and n_i denotes the count of the documents comprising of the word i). The system CLASSY [65] used *tf-idf* scores to rank sentences and is currently the best performing system on the dataset DUC 2004 in terms of *ROUGE-1* score.

Approaches like the cluster-based method [66] involved the semantic nature of a document (expressed in a natural language like English). The system MEAD [67] used document clustering to obtain salient words (highly similar to the topic) called centroids. These centroids further helped in the selection of sentences for summarization.

The methods, CLASSY and MEAD are similar to Google's *PageRank* algorithm [68] and Kleinberg's *HITS* algorithm [69]. However, the systems mentioned above proceed without considering the global frame of reference. Hence, an optimization method has been proposed based on Integer Linear Programming (ILP) which treats text summarization as a knapsack problem [70]. Authors also mentions the use of the ILP optimization for this task.

Graph-theoretic approaches, including *TextRank* [71] and *LexRank* introduced connections between the sentences (nodes) by an edge if the two sentences share common words. These approaches can visualize inter-document and intra-document similarities. Researchers mentioned the use of a graph-theoretic approach based on a weighted graphical representation of the documents obtained by topic modelling [72]. In another renowned classical approach, Latent Semantic Analysis uses Singular Value Decomposition (SVD) to find the important sentences in the document [73]. In KLSum [74], the summary is built using the criteria of Kullback-Lieber (KL) divergence. Biased centrality ranking called KP-centrality works by grouping semantically related sentences based on some known similarity measure. The size of the group that a sentence belongs to determines the rank of its importance [75]. The sentences' relationships are modelled as a graph by the graph-based ranking methods, which then use random walk algorithms to rate the sentences [71]. The unsupervised graph-based ranking is expanded upon by the CoRank model [62]. To assess sentences collectively, it can take into account data from numerous viewpoints, such as relationships between words and sentences and sentence-sentence interactions.

Other methods include HMM and CRF [76] based approaches, support vector regression-based methods [77], latent semantic analysis (LSA) and relevance measures. Authors mentions the use of a greedy approach to evaluate sentences for summarization [78]. An algorithm based on rhetorical statuses is used for summarizing scientific articles . More advanced techniques have also been introduced to measure the similarity of a sentence with the discourse by using synonyms and anaphoric relations. Researchers have also tried to integrated fuzzy logic and neural networks [80] in order to find a solution to the problem of text summarization. One specific area is covered by Parihar et. al. in which authors summarized the scientific papers based on citation and utilized game theory approach [79].

A document is represented as a graph in graph-based techniques, and word-word interactions are assessed using graph centrality measures such as PageRank and its derivatives. Words are represented by nodes in this model, which was first described as TextRank, and two words are connected by an edge if they co-occur in the given window size. PageRank is then used to rank nodes. Authors in 2008 enhanced TextRank to SingleRank by adding weights to edges, with the number of co-occurences between words serving as the weight. PositionRank illustrated the significance of capturing a word's position and frequency in a biased PageRank algorithm, preferring words that appeared earlier and more frequently in the text. There have also been attempts to extract key sentences that cover all of the document's important points. This was accomplished by either utilising a clustering-based approach to group words into specific themes or by employing the Latent Dirichlet Allocation (LDA) model to determine the topic distribution [81]. LDA-based algorithms necessitate training data, making the system corpus-dependent. were the first to employ clustering-based techniques in a graph. TopicRank, their suggested approach, clusters candidates based on the fraction of shared terms using hierarchical clustering [67]. In 2022 Jain et. al. proposed a methodology for keyword extraction using various fuzzy centrality measures and utilized localized tweets for experimentation [82]. Saxena et. al. proposed keygames, where authors applied game theory with word embeddings for keyphrase extraction [83]. Game theory has been applied to many other areas of natural language processing including rumour detection, query expansion, sentiment analysis etc [72, 80, 84, 85].

In this chapter, we have studied various techniques for natural language processing applications and identified real-world problems associated with these techniques. Various limitations have also been identified. we have addressed these challenges by proposing various novel models using soft computing approaches. In the next chapter, the proposed research presents a novel framework for text correction.

Chapter 3

Code-Mixed Hindi-English Text Correction

Social media facilitates sharing/exchanging ideas, information, thoughts, and other forms of expression via virtual networks and communities (Mayfield 2008). Social media activities include blogging, social networks (Twitter, Facebook), messaging, product reviews (Google play store, Amazon, Flipkart, Myntra), service reviews (Ola, Uber, 1mg, Urbanclap, TripAdvisor) and many more. In India, because of the widespread availability of internet access, the number of social media users has been growing. Social media has become one of the essential parts of daily internet usage in India. India is a multilingual country with 528 million Hindi & 125 million English speakers. A large number of users usually blend Hindi and English language during informal social media communication. This blended language is known as Hindi-English code-mixed language. The huge code-mixed data on social media is a rich source of information about the interest, behaviour and concern of users. The extracted information can be further utilized for applications such as search engines, data mining, sentiment analysis, cyber security, text-based forecasting, etc. But this enormous code-mixed data cannot be utilized in its original form. It comprises noisy elements such as multiple languages, misspelt word, slang terms, abbreviations etc.. The Hindi-English code-mixed text includes Devanagari Hindi, Roman Hindi and English languages. Supervised or unsupervised techniques need structured, clean data for text processing for satisfactory accuracy. For this, the work's main contribution is to pre-process the text, perform language identification, detection & correction of spelling errors in Roman Hindi, Devanagari Hindi and English words and make the text suitable for various NLP applications. Hindi is a complex language on its own as compared to English.

The rest of the chapter is organised as follows: Section 3.1 is for detailed proposed work. Section 3.2 illustrates the proposed work through examples, followed by the description of the experimental setup and results in section 3.3. Finally, Section 3.4 summarize the chapter.

3.1. Proposed Methodology

Large volume and heterogeneous unstructured data is available on social media, which has code-mixing, word spelling errors and cannot be effectively processed, queried, and analysed. In this work, the proposed method identifies the misspelt words and suggests correct alternative words for erroneous words. The overall architecture of the proposed work is shown in Fig.3.1.The proposed method consists of the following steps:

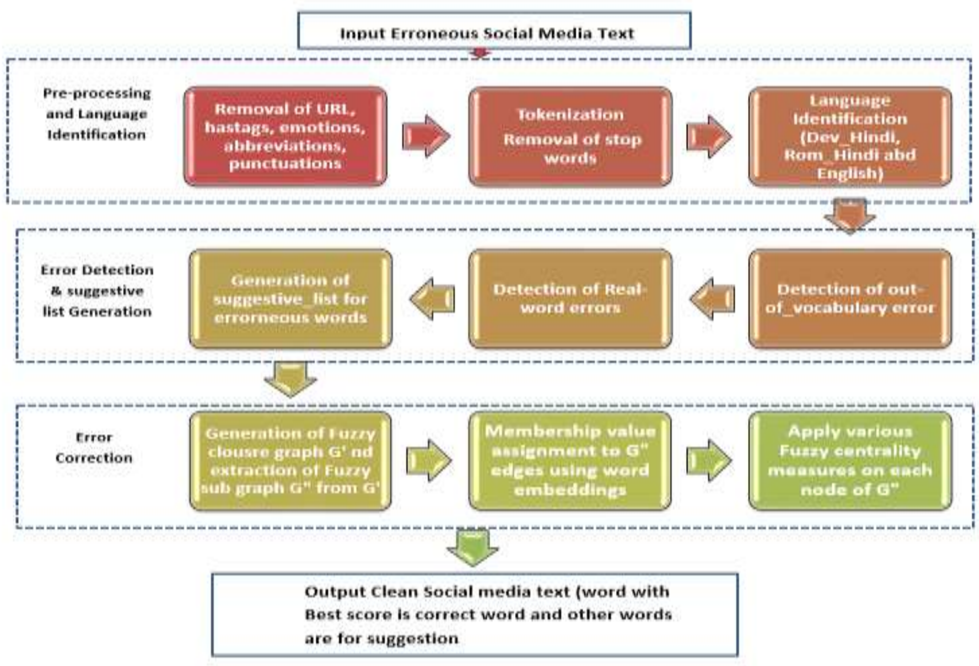
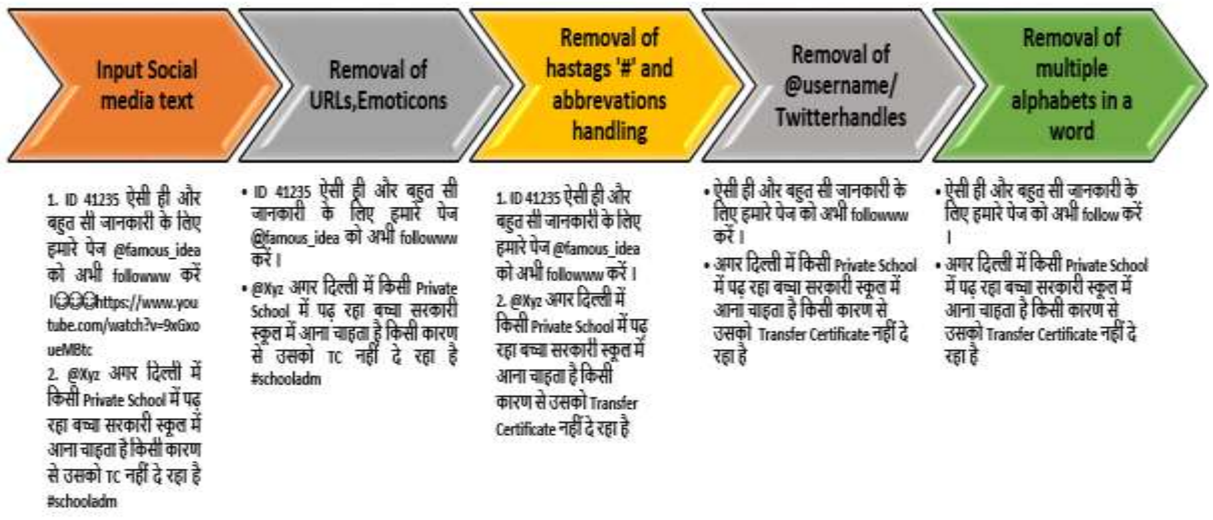

Input Erroneous Social Media Text
Pre-processing and Language Identification
Removal of URL, hastags, emotions, abbreviations, punctuations
Tokenization Removal of stop words
Language Identification (Dev_Hindi, Rom_Hindi abd English)
Error Detection & suggestive list Generation
Detection of out-of_vocabulary error
Detection of Real-word errors
Generation of suggestive_list for erroneous words
Error Correction
Generation of Fuzzy clousre graph G' nd extraction of Fuzzy sub graph G" from G'
Membership value assignment to G" edges using word embeddings
Apply various Fuzzy centrality measures on each node of G"
Output Clean Social media text (word with Best score is correct word and other words are for suggestion

Fig. 3.1. Overall Architecture for the Proposed Approach

3.1.1 Pre-Processing of Input Social Media Hindi-English Code Mixed Sentences

It is highly challenging to complete any further task on social media text because individuals utilise numerous symbols like "@" and "#," URLs, usernames, abbreviations like "ROFL," "TTYL," "LOL," "gotta," etc. As a result, thorough data cleaning is necessary to extract pertinent information. The pre-processing processes used to this model (shown in Fig. 3.2) are as follows:


Input Social media text
Removal of URLs,Emoticons
Removal of hastags '#' and abbrevations handling
Removal of @username/ Twitterhandles
Removal of multiple alphabets in a word

1. ID 41235 ऐसी ही और बहुत सी जानकारी के लिए हमारे पेज @famous_idea को अभी followww करें।😀😀😀https://www.youtube.com/watch?v=9xGxoueMBtc
2. @Xyz अगर दिल्ली में किसी Private School में पढ़ रहा बच्चा सरकारी स्कूल में आना चाहता है किसी कारण से उसको TC नहीं दे रहा है #schooladm

- ID 41235 ऐसी ही और बहुत सी जानकारी के लिए हमारे पेज @famous_idea को अभी followww करें।
- @Xyz अगर दिल्ली में किसी Private School में पढ़ रहा बच्चा सरकारी स्कूल में आना चाहता है किसी कारण से उसको TC नहीं दे रहा है #schooladm

1. ID 41235 ऐसी ही और बहुत सी जानकारी के लिए हमारे पेज @famous_idea को अभी followww करें।
2. @Xyz अगर दिल्ली में किसी Private School में पढ़ रहा बच्चा सरकारी स्कूल में आना चाहता है किसी कारण से उसको Transfer Certificate नहीं दे रहा है

- ऐसी ही और बहुत सी जानकारी के लिए हमारे पेज को अभी followww करें।
- अगर दिल्ली में किसी Private School में पढ़ रहा बच्चा सरकारी स्कूल में आना चाहता है किसी कारण से उसको Transfer Certificate नहीं दे रहा है

- ऐसी ही और बहुत सी जानकारी के लिए हमारे पेज को अभी follow करें।
- अगर दिल्ली में किसी Private School में पढ़ रहा बच्चा सरकारी स्कूल में आना चाहता है किसी कारण से उसको Transfer Certificate नहीं दे रहा है

Fig. 3.2. Steps of Pre-Text Processing

Elimination of URLs and Emoticons: The text extracted from social media consist of shared image or video URLs and emoticons, both URL's and emoticons are not helpful in proposed work so eliminated in this step.

Removal of hashtags and abbreviations handling: Users include hashtags in their posts to group them into categories that make it simple for other users to find and follow content on a certain subject. In pre-processing, hashtags are also deleted. Next abbreviations are converted in to their original words for better understanding of context, for instance OMG- Oh my god, BRB- be right back etc.

Removal of username/Twitter handles: Many times in text from social networking sites contain the symbol '@' or ID's trailed by a username. These usernames should be removed. In this work, this is done using regex.

Elimination of multiple alphabets in a word: To emphasize the particular word, social media users extend word by adding multiple characters for instance, I am sooo happpyyyy, I feel greatttt etc. so we normalized these words in pre-processing step.

Tokenization: Tokenization is the method of locating tokens or fundamental units that do not require further breakdown. The terms in the tweet are each treated as a token. For instance, 'Draw the lines' becomes ['Draw', 'the', 'lines'].

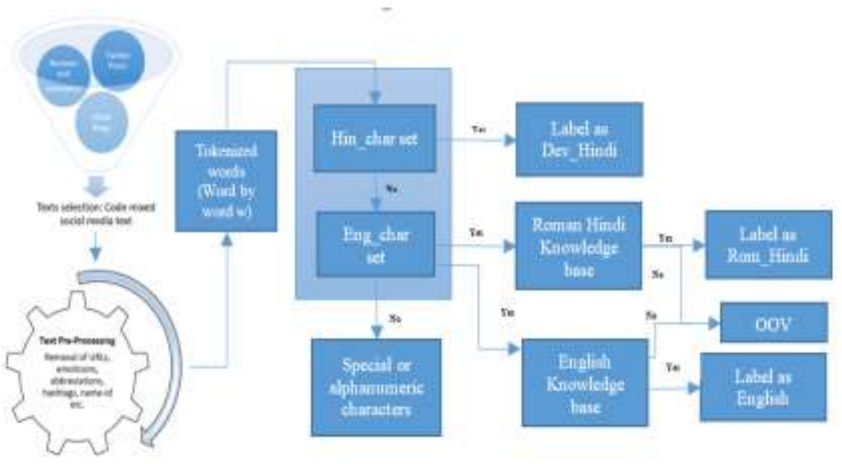

Texts selection: Code mixed social media text
Text Pre-Processing
Removal of URLs, emoticons, abbreviations, hashtags, name-id etc.
Tokenized words (Word by word w)
Hin_char set
Yes
Label as Dev_Hindi
No
Eng_char set
Yes
Roman Hindi Knowledge base
Yes
Label as Rom_Hindi
No
OOV
Yes
No
English Knowledge base
Yes
Label as English
No
Special or alphanumeric characters

Fig.3.3. Steps for Language Identification Process

3.1.2 Language Identification

Language identification is an essential step in code mixed language where users have been using more than one language to express their thoughts. Social media text is highly informal and can be written in English, Devanagari Hindi(Dev_Hindi), Roman Hindi. To process this kind of data, it is necessary to identify the language. We used word-based language identification to differentiate between Dev_Hindi, Rom_Hindi and English words. Fig. 3.3 and algorithm 3.1 show the proposed language identification process in detail.

Collected social media data is first pre-processed, and then tokenized words one by one goes for checking in Hindi character sets. We are using the Hindi character set to identify language rather than passing it to the whole dictionary because it is computationally fast to check in the 40-character set rather than millions of words. If a word has Hindi characters, it is labelled as Devanagari Hindi (Dev_Hindi). If a word has English characters, it can be a Roman Hindi word or an English word. Now next is to check the word in Roman-Hindi knowledge bases and English knowledge base. If the word is available there, it is marked as Rom_Hindi, English word respectively else, it is an Out of Vocabulary (OOV) word which is handled in next section. And if the word is neither in Hindi character set nor in English character set, it has special or alphanumeric characters which is discarded.

3.1.2.1 Resources for Roman Hindi Vocabulary

This work uses two available Rom_Hindi knowledge bases

1. By IIT Bombay, XLIT-crowd transliteration pairs. The data for this knowledge base is mined via crowdsourcing on Amazon Mechanical Turk (AMT). This dataset comprises 14,919 Roman-Devanagari Hindi word pairs.

2. FIRE track public transliteration pairs- This dataset comprises 30,823 Hindi words in Devanagari and Roman. These pairs are collected using the alignment of the Hindi song's lyrics.

Algorithm 3.1: Language Identification

```
Input: Pre-processed code-mixed text.
Output: Identification of language with proper tag/label
1.for each word, ‘w in input text
2.     if w is available in Hin_char set then
3.       Assign tag ‘Dev_Hindi’ to w
4.     if w is in Eng_char set then
5.        if w is in Roman-Hindi knowledge base
          then
6.            Assign tag ‘Rom_Hindi’ to w
7.       else
8.            Assign tag ‘OOV’ to w
9.       if w is in English knowledge base  then
10.           Assign tag ‘English’ to w
11.      else
12.           Assign tag ‘OOV’ to w
13.    if w has special or alphanumeric character
14.      Discard w
```

3.1.3. Detection of Spelling Errors and Generation of Suggestion List

Now, we have pre-processed, language tagged words and next step is to detect erroneous words with suitable suggestions. For English language identification, the proposed method made use of the Python Package Enchant (Kelly 2016). Enchant is a spell checker that provides suggestions for possible erroneous words in text. Enchant also work with the Devanagari script, but our data also includes Romanised Hindi words. In addition to this our main concern is to handle both type of errors i.e. real-word and non-word errors in Devanagari and Roman script, which is not possible for Enchant library. In our proposed work, we are using Enchant for English language identification and spelling correction only. For roman Hindi we used comprehensive knowledge bases. And for Devanagari, we have proposed an algorithm for detection and suggestion words. Overall structure for detection and suggestion list is shown in Fig. 3.4. Language based discussion for error detection is discussed below.

Table 3.1. Instances for Real-word errors in Rom_Hindi text (Errors are highlighted)

Rom_Hindi sentences with highlighted erroneous words	Meaning of correct word	Meaning of the erroneous word
maine khan kha liya	Khaana -food	Khan- A caste
Mujhe ab dari nahi lagta	Dar- fear	Dari- Carpet
Sudh bhasha ka upyog jaruri hai	Shudh - pure	Sudh - consciousness

For Roman Hindi Words

In literature till now, there is no spelling error detection and correction system for Rom_Hindi words, simply because there are no standards defined for Rom_Hindi script. Users generally tend to write different spellings according to their speaking style, available time or normally users follow the trend which makes them look more cool or trendy. This is commonly observed users miss vowels while writing for example “scl for school”, “ghr for ghar” etc. Real word errors in Rom_Hindi script can create a massive blunder while understanding the context of the script. For instance, maine khan kha liya here correct word is khaana “khaana is food” and “khan is a caste”. More instances are shown in Table 3.1, we have performed steps to identify and correct real-word spelling errors in Rom_Hindi text also. Input Rom_Hindi words are checked in knowledge bases (XLIT and FIRE) if word is not available then levenstien distance algorithm is used to identify similar words (suggestive List), Else there is no OOV error but possibility of real-word error is still there. In this case all the open class words (nouns, adjectives and adverbs) of the sentence are passed through levenstein algorithm to get suggestive List.

For Devanagari Hindi Words

For detection of erroneous words, input pre-processed Dev_Hindi words are passed through Hindi dictionary. We have used Hindi WordNet for Hindi Dictionary. It has around 70,000 unique tokens. Our aim- is to detect both Non-word (OOV) and real-word errors in the text.

So, after the dictionary lookup, if the word is not available in the dictionary, it goes for the Wikipedia validation. Wikipedia validation is an essential step. Many new words maybe not available in Hindi WordNet dictionary which can be found on Wikipedia. If the word is not found on Wikipedia, then it is undoubtedly a non-word error. In case of real-word errors, probability of occurrence (TF-IDF) is calculated. The tf-idf determines how considerably the word is related to a document by weighting each word. Frequency of the correct word for instance “Cake-piece” would be high than the incorrect word “cake-peace” in the document.

For suggestive List, we have utilized data available in literature. Jain and Jain (2018) analyzed the erroneous data and identified the most frequently occurred spelling errors by users. It is observed that user performs maximum errors in case of phonetically similar characters. The phonetically similar character set for Dev_Hindi can be seen in Table 3.2. The table have four different groups which are phonetically same set of matras, consonants, vowels and symbols. On the basis of this table, the system generates a suggestive List using algorithm 3.2.

3.1.4. Generation of Fuzzy Graph Based on the Suggestive List Using Hindi WordNet

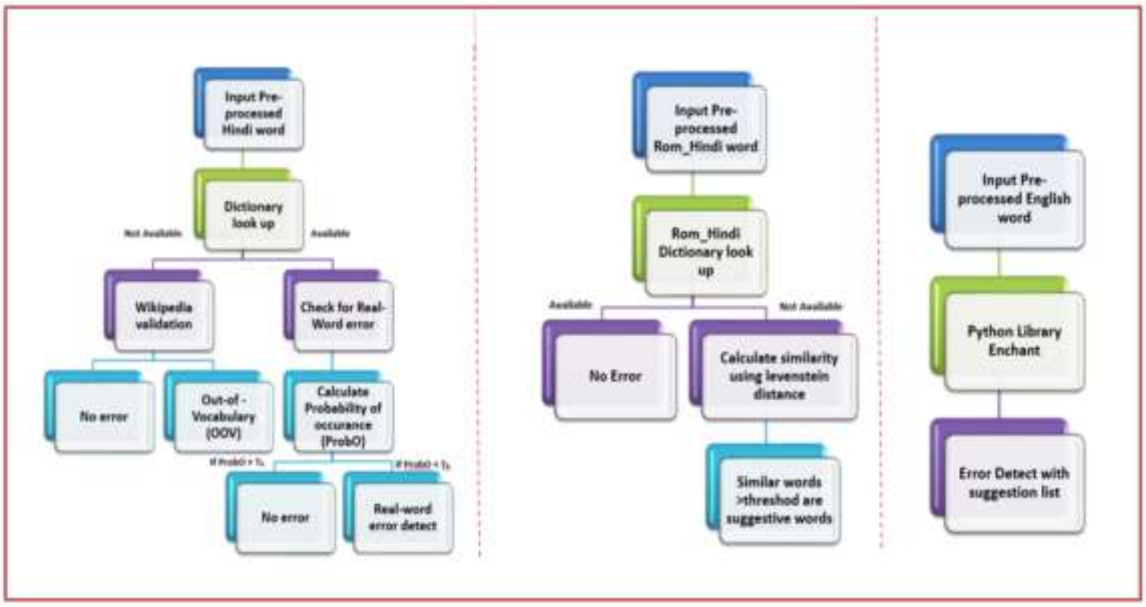
After the pre-processing, language identification, detection of erroneous word and generation of suggestive List, next is to translate each word in same language that is Devanagari Hindi to get better context between words. After this, next is to generate fuzzy graph using suggestive list words. Let the user have entered the sentence S, $S =$

$(w_1, w_2, w_3, \ldots, w_n)$, n is the number of words in S, For the i^{th} erroneous word w_i, suggestive List is $(w_{i1}, w_{i2}, w_{i3}, \ldots, w_{im})$ where m is the number of words in list for suggestions.

To generate fuzzy graph from suggestive list, this work utilized four types of word relations from Hindi WordNet which are hypernymy, hyponymy, meronymy, and holonymy. Algorithm 3.3 shows the steps involved in fuzzy graph generation. Input of this algorithm is a list of the suggestive list for all the words w_i and the output is a fuzzy graph G'which contains all words of suggestive list as nodes and all other nodes connection between edges between different suggestive list.

3.1.5 Extraction of fuzzy sub graph G'' from the fuzzy graph G'

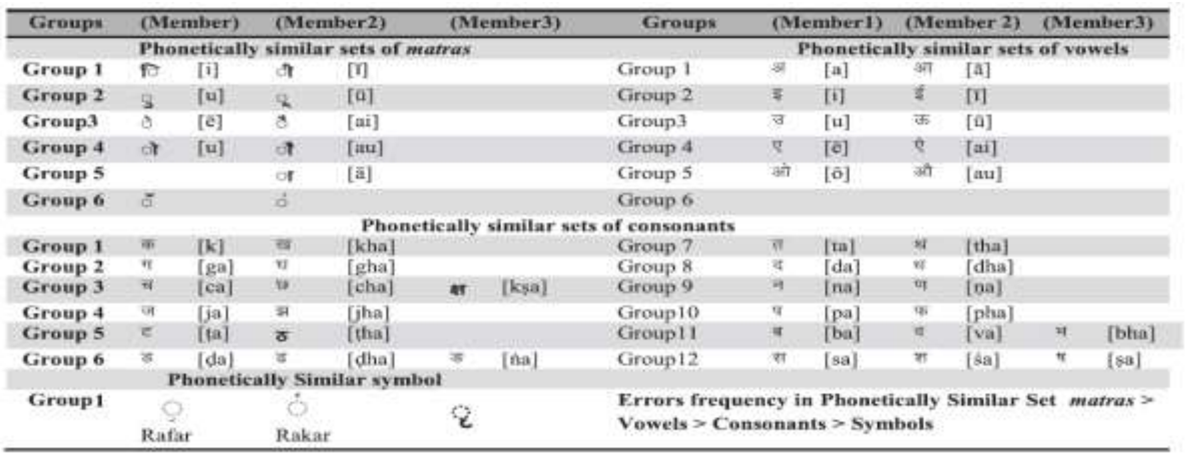
Now we have our closure fuzzy graph G' and we need to apply fuzzy centrality measures on nodes to find the best suitable suggestive from suggestive_list for the erroneous word. But the fuzzy graph G' is too big in size and also have some insignificant nodes which may cause an error in result. So, we apply Depth First Search (DFS) on our graph G' up to a certain depth D=5 (experimentally determined) to extract a sub graph G". Using Algorithm 3.4, subgraph G^" is generated.



Input Pre-processed Hindi word
Dictionary look up
Not Available
Available
Wikipedia validation
Check for Real-Word error
No error
Out-of -Vocabulary (OOV)
Calculate Probability of occurance (ProbO)
If ProbO > T1
If ProbO < T1
No error
Real-word error detect
Input Pre-processed Rom_Hindi word
Rom_Hindi Dictionary look up
Available
Not Available
No Error
Calculate similarity using levenstein distance
Similar words >threshod are suggestive words
Input Pre-processed English word
Python Library Enchant
Error Detect with suggestion list

Fig.3.4. Detection of spelling errors and Suggestive List generation steps for all languages

Table 3.2. Phonetically Similar Character Set in Hindi



Groups	(Member)	(Member2)	(Member3)	Groups	(Member1)	(Member 2)	(Member3)
	Phonetically similar sets of *matras*				**Phonetically similar sets of vowels**		
Group 1	ि [i]	ी [ī]		Group 1	अ [a]	आ [ā]	
Group 2	ु [u]	ू [ū]		Group 2	इ [i]	ई [ī]	
Group3	े [ē]	ै [ai]		Group3	उ [u]	ऊ [ū]	
Group 4	ो [u]	ौ [au]		Group 4	ए [ē]	ऐ [ai]	
Group 5		ा [ā]		Group 5	ओ [ō]	औ [au]	
Group 6	ँ	ं		Group 6			
			Phonetically similar sets of consonants				
Group 1	क [k]	ख [kha]		Group 7	त [ta]	थ [tha]	
Group 2	ग [ga]	घ [gha]		Group 8	द [da]	ध [dha]	
Group 3	च [ca]	छ [cha]	क्ष [kṣa]	Group 9	न [na]	ण [ṇa]	
Group 4	ज [ja]	झ [jha]		Group10	प [pa]	फ [pha]	
Group 5	ट [ṭa]	ठ [ṭha]		Group11	ब [ba]	व [va]	भ [bha]
Group 6	ड [ḍa]	ढ [ḍha]	ङ [ṅa]	Group12	स [sa]	श [śa]	ष [ṣa]
	Phonetically Similar symbol						
Group1	र् Rafar	्र Rakar	ृ	**Errors frequency in Phonetically Similar Set *matras* > Vowels > Consonants > Symbols**			

ALGORITHM 3.2: Suggestive list Generation for Dev_Hindi Words

---

```
Input: Erroneous Word $w_i$ entered in sentence $S = (w_1, w_2, w_3 \ldots w_i \ldots w_n)$
Output: The List of suggestion words called Suggestive_List ($w_{i1}, w_{i2} \ldots w_{ij} \ldots . w_{ir}$)
1.for each character in $w_i$
2.     if char = matra
3.        then replace with phonetically same category member of matra// please
          refer to Table 3.2
4.        Dictionary lookup()
5. for each character in $w_i$
6.     if char = vowel
7.        then replace with phonetically same category member of vowel// please
          refer to Table 3.2
8.        Dictionary lookup()
9. for each character in $w_i$
10.    if char = consonant
11.       then replace with phonetically same category member of consonant//
          please refer to Table 3.2
12.       Dictionary lookup()
13. for each character in $w_i$
14.    if char = half-consonant && next char != matra
15.       then replace every consonant with its half consonant
16.       Dictionary lookup()
```

```
17.   if char = half-consonant
18.      then replace half consonant with its full consonant
19.      Dictionary lookup()
20. for each character in $w_i$
21.   if char = rafar or rakar
22.      then Dictionary lookup() after interchange
23. for each character in $w_i$
24.   if char = anuswara or anusika
25.      then Dictionary lookup() after interchange
26. Perform transposition, substitution , insertion and deletion operations on character
27.   Dictionary lookup()
28. Return Suggestive_list
```

ALGORITHM 3.3: Fuzzy graph generation

$G^{'}$ is an empty graph, SL_i is suggestion list i.e, $(w_{i1}, w_{i2}, w_{i3}, \dots, w_{im})$ for i^{th} erroneous word w_i.

```
1. $Let\ l_i = SL_i, m = |l_i|$
2. for k = 1 to m do
3.    if ($l_i$[k] $\notin G^{'}$)
4.   add_node($l_i$[k])
5.  Let synset = list of synsets of $l_i$[k], l = |synset|
6.  for j = 1 to l do
7.     lem = lemma names of synset[j], n = |lem|
8.      for i = 1 to n do
9.          if($lem[i] \notin G'$)
10.            add_node(lem[i])
11.     if (lem[i].hypernymy())
12.       synset’ = lem[i].hypernymy()
13.       Repeat steps 5-12 synset’
14.     Else goto 15
15.    Repeat steps 11-14 for other semantic relations like hyponymy, meronymy and holonymy
```

ALGORITHM 3.4: Sub Graph $G"$ generation.

G'' is an empty graph, SL_i is suggestive list i.e, $(w_{i1}, w_{i2}, w_{i3}, \dots, w_{im})$ for i_{th} headword w_i.

```
1.$Let\ l_i = S_{Li}, m = |l_i|$
2. for k = 1 to m
3.   if ( $l_i[k] \notin G'$)
4.      $add\_node(l_i[k]$
```

5. for $l_i[k] \in SL_j, 1 \leq j \leq n,$ where n is number of headwords the system performs a depth-first search (DFS) on the WordNet graph G' and whenever the system finds a $(node \in SL_j)$ such that $(node \neq l_i[k])$ along a path $node, n_1, \ldots, n_k, node'$ of $length \leq D$, the system will insert all the transitional nodes and edges on the path from v to v' in the sub graph G''.

3.1.6. Automatic Computation of Fuzzy Weight and Fuzzy Centrality Measures

After formation of subgraph G", each edge's fuzzy weight, that measures the degree of relationship between the two vertices, is determined automatically. This is accomplished by converting each vertex into a vector and calculating how similar the vectors are to one another. This score can be calculated using Word2Vec embeddings [86]. For training of word2vec embeddings for the proposed approach, Gensim2 in Python is used. We take the window size to be 5, and the word vector's size to be 200. Every utterance with a frequency below two is regarded as noise. .The most popular membership function, cosine similarity, is used in Word2Vec word embedding models. Using this paradigm, the cosine similarity between the vectors of two words can be used to determine how similar two words are. The cosine angles of the two vectors are multiplied to determine the cosine similarity.

Implement fuzzy centrality metrics such as fuzzy degree, fuzzy closeness, fuzzy page rank, and fuzzy betweenness, on each element of suggestive list in the sub graph G' and then the words with best score of the centralities are the correct words by the system for every erroneous word.

3.2. Illustrations Through Examples

In the following section, the above method is explained through two examples. Sample sentence taken from social media for the first example is: “@harjinder13 farmers का समान करना हमारा ध्रम है #farmerprotest #kisan_mazdoor”. After pre-processing steps discussed in section 4.1, new input sentence is “farmers का समान करना हमारा ध्रम है”. Applying error detection step, erroneous words are: “समान” which is a real-word error and “ध्रम” which is an out of vocabulary or non-word error. As described above, the first step is generation of suggestive lists for erroneous words of the input sentence. The stop words in the sentence – “का”, “करना”, “है” are removed. Using algorithm 3.2, suggestive List for Dev_Hindi word is generated as shown in Table 3.3. All the words are translated into Dev_Hindi i.e. farmers is converted in किसान to get better context between words. Suggestive List and input sentence words are now seed words for graph generation. Seed words for the graph generations are: किसान ,समान, सामान, सम्मान, कमान, समाज, धर्म, क्रम, ड्रम, भ्रम, श्रम. Following this, a fuzzy graph, G', is generated using algorithm 3.3 as shown in Fig. 3.5. Next subgraph is shown in Fig. 3.6. with fuzzy edge weight assigned using word2vec embeddings. After applying fuzzy centrality measures of subgraph results are shown in Table 3.4, and bold values shows the correct word for the erroneous word.

Table 3.3. Suggestive List for illustration 1.

Input word = “समान”	Input word = “ध्रम”
समान “*samāna*” (equal)	धर्म “*ḍharma*” (religion)
सामान “*sāmāna*” (goods)	क्रम “*krama*” (sequence)
सम्मान “*sammāna*” (respect)	ड्रम “*ḍrama*” (drum)
समाज “*samāja*” (society)	भ्रम “*bhrama*” (confusion)
	श्रम “*śrama*” (labor)

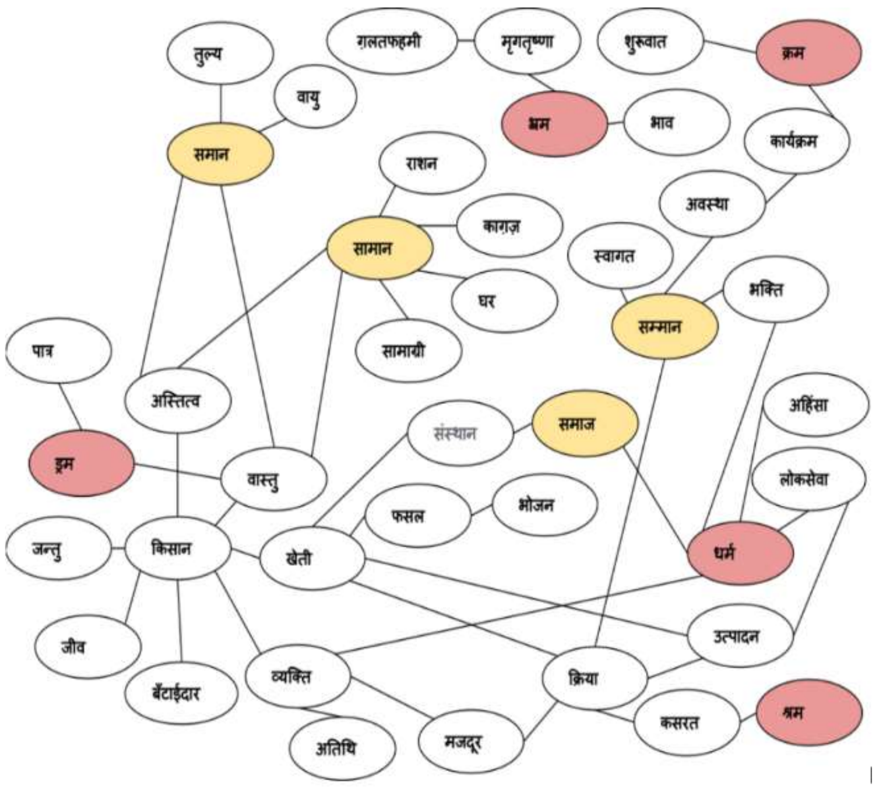


Fig. 3.5. Fuzzy graph G' for illustration 1

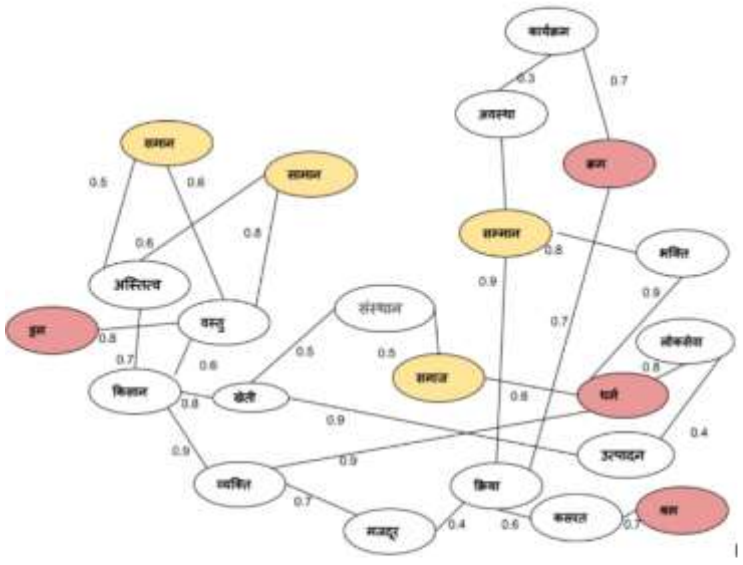


Fig. 3.6. Fuzzy Subgraph G" for illustration 1

Table 3.4. Results for illustration 1

Nodes	Weighted Degree	Authority	Hub	Pageranks	Eigencentrality	Closness Centrality	Betweeness Centrality
सामान	2.4	0.199211	0.199609	0.03258	0.391009	0.276316	0.666667
समान	2.6	0.199211	0.199609	0.034729	0.391009	0.276316	0.666667

सम्मान	**4.8**	**0.333347**	**0.332952**	**0.075698**	**0.746725**	**0.456522**	**85**
समाज	1.6	0.110499	0.110375	0.031534	0.26755	0.283784	3.833333
धर्म	**5.8**	**0.225598**	**0.225781**	**0.075849**	**0.527255**	**0.362069**	**33.5**
क्रम	2.8	0.097803	0.097768	0.041157	0.25483	0.287671	4.5
ड्रम	1.6	0.105256	0.105468	0.023999	0.207547	0.269231	0
श्रम	1.4	0.026749	0.026768	0.026209	0.078469	0.223404	0

Illustration 2:

Sample sentence: “Omg looks yummmmm, healthy khaan bhi itna tasty ho sakta hai.. <3 #loveforfood #stayfit

” . After pre-processing steps discussed in section 4 , new input sentence is “Oh my god looks yum, healthy khaan bhi itna tasty ho sakta hai” Using error detection step erroneous word is: “khaan” is a real-word error. As described above, the first step is generation of suggestive lists for erroneous words of the input sentence. The stop words in the sentence are removed and the remaining words are: Healthy, khaan, tasty. Using algorithm 3.2, suggestive List for Dev_Hindi word is generated. All the words are translated into Dev_Hindi i.e. Healthy, khaan, tasty is converted into Devanagari get better context between words. Suggestive List and input sentence words are now seed words for graph generation. Seed words for the graph generations are: खाना, खोना ख़ान. Following this, a fuzzy graph, G', is generated using algorithm 3.3 as shown in Fig. 3.7. The next subgraph is shown in Fig. 3.8. with fuzzy edge weight assigned using word2vec. After applying fuzzy centrality measures of subgraph results are shown in Table 3.5, and bold values shows the correct word for the erroneous word.

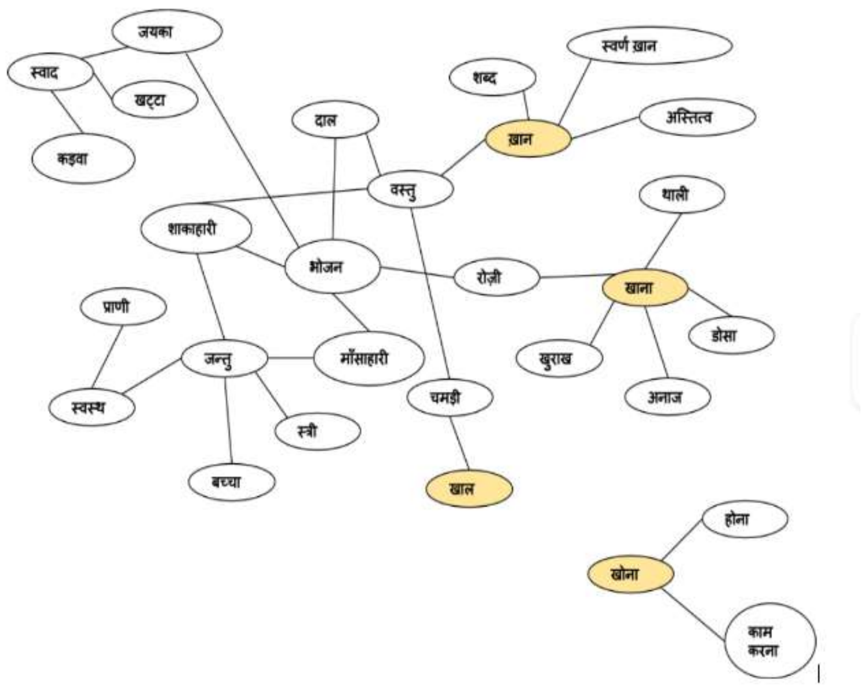


Fig. 3.7. Fuzzy graph G' for illustration 2

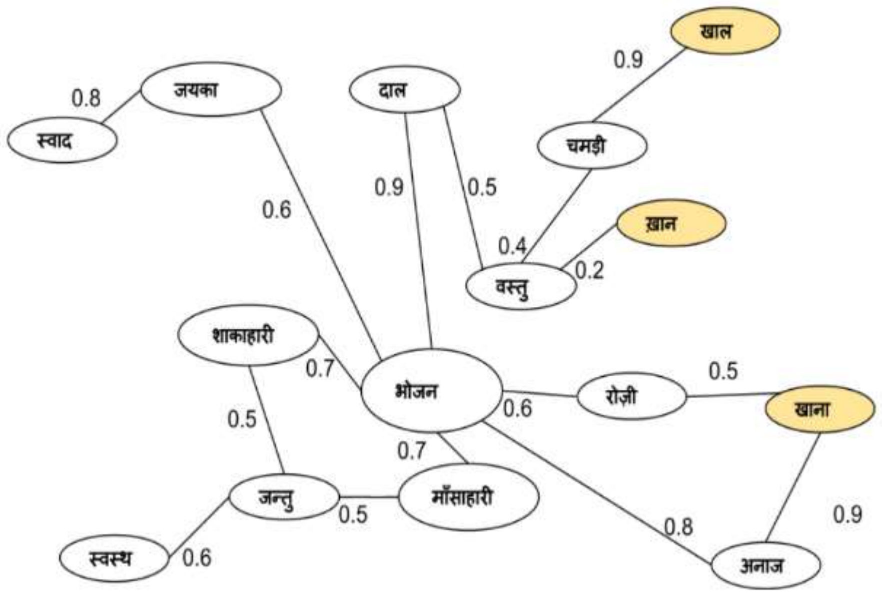


Fig. 3.8. Fuzzy Subgraph G^" for illustration 2

Table 3.5. Results for illustration 2

Nodes	Weighted Degree	Authority	Hub	PageRank	Eigen centrality	Closeness centrality	Betweenness centrality
ख़ान	0.4	0.03684	0.053562	0.02074	0.115362	0.269231	0
खोना	1.8	0.021886	0.015053	0.057047	0.067145	0.222222	0
खाना	**2.8**	**0.241674**	**0.166222**	**0.06195**	**0.331255**	**0.285714**	**0.5**

3.3. Experimental Setup, Evaluation and Results

This section performs several experiments to test the efficiency of the proposed method. This section includes evaluation matrices, test resources, experiment results on various datasets and sentiment analysis applications.

Evaluation matrices: The metrics employed to evaluate the effectiveness of the proposed approach are: Recall (R), Precision (P), and F-measure (F).

Precision (P), and F-measure (F).

$$R = \frac{Number\ of\ precise\ spelling\ detections/corrections\ done}{Total\ number\ of\ incorrect\ words}$$

$$P = \frac{Number\ of\ precise\ spelling\ detections/corrections\ done}{Total\ number\ of\ detections/corrections}$$

F-measure is the harmonic mean of recall and precision and can be calculated by

$$F = \frac{2PR}{P+R}$$

3.3.1. Experimental Results on Various Datasets

For evaluation, the proposed method is applied to sentences (approx. 38,000) extracted from various social media sources including Twitter, YouTube comments, Instagram comments, blogs and WhatsApp chat. For tweets scrapping, python package twitterscraper is used. Total 10,356 tweets from January 6, 2022 to January 19, 2022 were extracted using query movie and location India. There were 4156 tweets written in English and 2754 written in Devanagari Hindi. And a total of 3446 were English-Hindi code-mixed tweets. The second dataset was taken from YouTube (YT) comments containing 5034 sentences, third was extracted from Instagram comments comprising 7979 sentences, fourth from blogs, 9786 sentences. For English slang terms, we use the NoSlang dictionary6. For English, we use the dictionary developed by Han et al. consist of 41,181 tokens for micro-blog normalization task [87]. Limited data of approx. 5000 messages from WhatsApp have been collected through personal relations (students, colleagues, parents, relatives).Table 3.6 & 3.7 describe the dataset and words in each category.

The results are analysed differently for real-word and OOV (non-word) error for different erroneous social media sentences. The threshold value L is the parameter used to describe the results for both errors (non-word and real-word) which is defined as the maximum level in the suggestive list up to which the correct word can be found.. Table 3.8. shows the results of Dev_Hindi non-word error & real word. It can be seen from the table that the system starts giving significant positive results from level 1(L=1) with the precision and recall of ≈0.85 and ≈ 0.65 respectively. The method proposed for generation of suggestive list is providing best-suited suggestive for starting positions. System's performance can be improved significantly if we increase the value of level L at the cost of computation efficiency. Table 3.8 shows the increasing value of precision and recall for increasing value of L. For L=5 it shows recall value of ≈ 0.9 that is 90% of errors are corrected by the system for non-word errors.

We can see the results for real-word errors in Table 3.8. We have obtained different precision and recall for different depths of L. We can see precision and recall of 0.89 and 0.67 for L=5 respectively. It is visible that our system is correcting 70% of real-word error for L=5.

There is no value shown of precision and recall for L=1 as the error is real-word so it is available in the dictionary hence, the error word is present at L=1. It is clearly visible that precision and recall values are much better for non-word error as compared to real-word errors. Because handling real-word error is much more difficult as they are available in the dictionary.

Table 3.9 is for Rom_Hindi words, the results are not as good as that of Dev_Hindi correction system because of complexities. There is no work for both real-word as well as non-word errors in social media text. Same experiments are performed for

Rom_Hindi words, but there is no value of L this time because, now all the suggestive words have equal priority. And system archived maximum 87% recall

for non-word and 66% recall for Real-word errors. Graphical representation of results is shown in Fig. 3.9.

While testing our system, we came across certain hindrances and limitations. The words have been taken from Hindi WordNet which has around 70,000 unique words only. As Hindi is a very vast language and there is no complete dictionary available, errors occur due to false correction. There are words which are correct and are placed at their right position but still considered as non-word error as they are not present in the Hindi WordNet. Therefore, the system doesn’t include them in the suggestive list. Then the system replaces the correct word with one word from the suggestive list. For example, the word रिहा “riha” (free), which is a real-word but is not present in dictionary. Therefore, the system will replace this word with some other word of suggestive list This is a common problem with spelling error detection and correction systems while handling new words. Another challenge while testing our system occurred when a real-word became a stop word. There are some words which convert to stop word while processing such as तोल “tol” (measure) which after deletion of the last character ल becomes तो “toh” (So). As the word becomes a stop word, it cannot be corrected as system will consider it as a stop word and will ignore it. While generating our closure graph G', the system used hypernymy, hyponymy, meronymy, holonymy and similarity of the words of suggestive list but if we encounter an adjective or adverb then we need to skip these words as these semantic relations are not applicable on adjective and adverb. Table 3.10 shows the comparative results with Etoori et al. [88] and n-gram based Viterbi algorithm by jain et. al. [21]. Proposed approach clearly outperforms the both state-of-art methods.

Table 3.6. Dataset Description

	Total sentences	**English sentences**	**Dev_Hindi sentences**	**Code-mixed sentences**
Tweets	10,356	4156	2754	3446
YouTube	5034	1220	1357	2457

Instagram	7979	3785	1834	2360
Blogs	9786	3564	4678	1444
WhatsApp	4834	1869	1348	1617

Table 3.7. Code-mixed dataset description

Code-mixed sentences (Language identification)			
Resources	**Dev_Hindi words**	**English words**	**Rom_Hindi words**
Tweets	1258	1140	1048
YouTube	632	974	851
Instagram	782	845	733
Blogs	541	425	478
WhatsApp	654	358	605

Table 3.8. Results for OOV and Real-word (Dev_Hindi) on different social media platforms

		OOV (Dev_Hindi)			Real-Word error (Dev_Hindi)		
	Level L	Recall	Precision	F-measure	Recall	Precision	F-measure
Tweets	*L=1*	0.632	0.848	0.724	-	-	-
	L=2	0.689	0.860	0.765	0.445	0.685	0.539
	L=3	0.715	0.888	0.792	0.468	0.714	0.565
	L=4	0.790	0.895	0.839	0.555	0.776	0.647
	L=5	0.833	0.915	0.872	0.598	0.810	0.688
	L>5	0.897	0.948	0.921	0.645	0.845	0.731
YouTube comments	*L=1*	0.657	0.865	0.747	-	-	-
	L=2	0.667	0.878	0.758	0.459	0.654	0.539
	L=3	0.736	0.898	0.809	0.478	0.687	0.564
	L=4	0.790	0.910	0.846	0.543	0.798	0.646
	L=5	0.838	0.916	0.875	0.605	0.827	0.699
	L>5	0.904	0.953	0.928	0.655	0.898	0.757

WhatsApp	*L*=*1*	0.587	0.888	0.707	-	-	-
chat	*L*=*2*	0.667	0.890	0.763	0.483	0.672	0.562
	L=*3*	0.795	0.898	0.843	0.492	0.686	0.573
	L=*4*	0.812	0.903	0.855	0.578	0.707	0.636
	L=*5*	0.836	0.915	0.874	0.597	0.789	0.68
	L>*5*	0.887	0.957	0.921	0.632	0.834	0.719
Instagram	*L*=*1*	0.702	0.855	0.771	-	-	-
	L=*2*	0.719	0.860	0.783	0.354	0.678	0.465
	L=*3*	0.750	0.889	0.814	0.412	0.759	0.534
	L=*4*	0.790	0.890	0.837	0.466	0.789	0.586
	L=*5*	0.833	0.915	0.872	0.523	0.813	0.637
	L>*5*	0.897	0.976	0.935	0.634	0.898	0.743
Blogs	*L*=*1*	0.682	0.812	0.741	-	-	-
	L=*2*	0.709	0.834	0.766	0.467	0.602	0.526
	L=*3*	0.795	0.856	0.824	0.478	0.695	0.566
	L=*4*	0.790	0.878	0.832	0.598	0.789	0.68
	L=*5*	0.883	0.888	0.885	0.613	0.820	0.702
	L>*5*	0.901	0.946	0.923	0.678	0.889	0.769

Table 3.9. Results for OOV and Real-word (Rom_Hindi) on different social media platforms

	OOV (Rom_Hindi)			Real-Worderror (Rom_Hindi)		
	Recall	Precision	F-measure	Recall	Precision	F-measure
Tweets	0.743	0.646	0.691	0.546	0.745	0.63
YouTube comments	0.798	0.765	0.781	0.645	0.687	0.665

WhatsApp chat	0.622	0.785	0.694	0.567	0.756	0.648
Instagram	0.849	0.825	0.837	0.623	0.789	0.696
Blogs	0.874	0.865	0.869	0.664	0.635	0.649

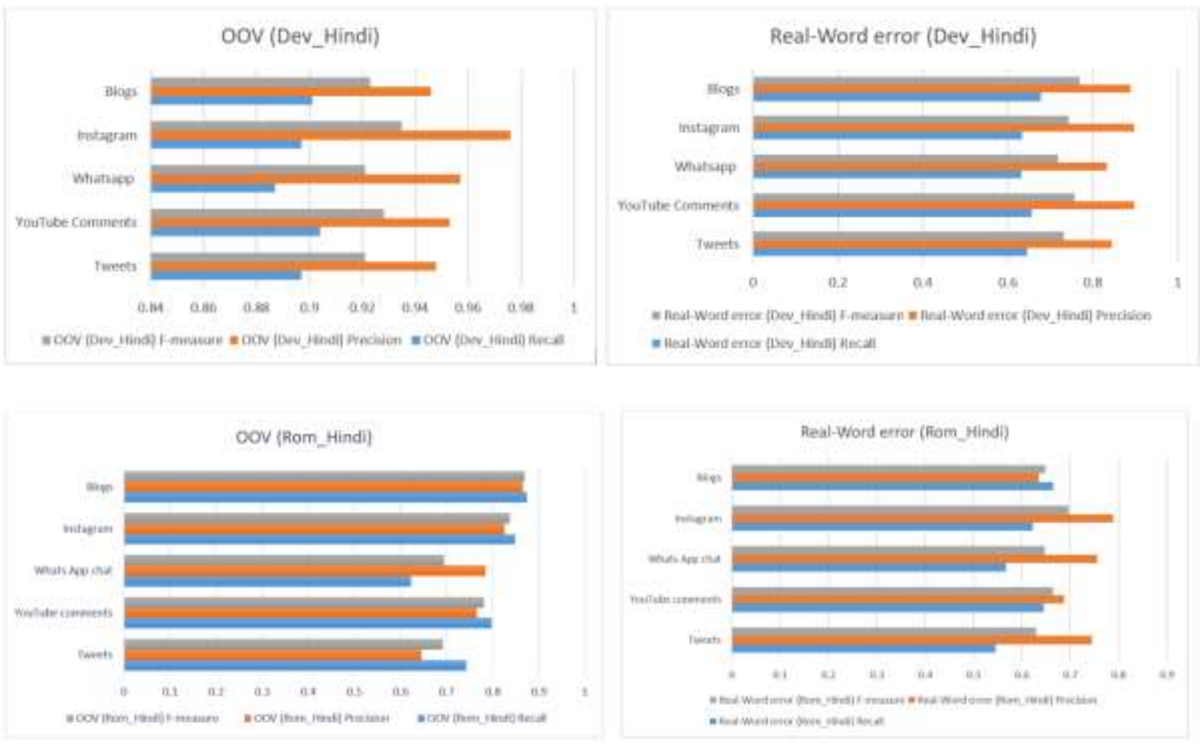


Fig. 3.9. Graphical Representation for results including OOV (Rom_Hindi), Real-word errors (Rom_Hindi), OOV (Dev_Hindi), Real-word errors (Dev_Hindi)

Table 3.10. Comparative results with state-of-art Methods

Devnagari Hindi	Evaluation matrices	Jain et.al 2018	Etoori et. al. 2018	Proposed methodology
Non-word Errors	Precision	0.91	0.95	0.94
	Recall	0.82	0.83	0.84
	F-measure	0.86	0.89	0.93
Real-Word Errors	Precision	0.80	0.73	0.85
	Recall	0.59	0.55	0.66
	F-measure	0.68	0.63	0.73

3.4. Summary

In this chapter, the authors proposed a methodology to normalize the erroneous code-mixed social media text. The proposed work is the first work that handles both errors (Real-word and Non-word) in Devanagari Hindi, Roman Hindi and English languages occur together in a code-mixed text. This work demonstrated that language identification and spelling correction of social media text is a crucial step before any mining/processing the text. In this work, authors pre-processed the text, including removing punctuations, abbreviations and many more informal language complexities. This work also proposed a system that performs language identification in Hindi-English code-mixed text. We found the concept of the fuzzy graph, fuzzy graph centrality measures and word embedding (word2Vec) utilized for spelling correction can handle social media informal text efficiently. Various experiments are performed on different social media platform's dataset collected from Instagram, Twitter, WhatsApp. The experimental results showed that the proposed system corrects out-of-vocabulary words as well as real-word spelling errors with a maximum recall of 0.90 and 0.67, respectively for Dev_Hindi and 0.87 and 0.66, respectively for Rom_Hindi. The proposed approach is also applied to the sentiment analysis task to analyse the importance of the proposed method (text standardization).

In the future, this work can be extended for different code-mixed languages. In social media text Devanagari or English is also mixed with other script, such as Indian language (Bhojpuri, Haryanvi, Punjabi etc.) and Foreign languages (Spanish, German many more). For this, need to have a corpus and knowledge base for words of different languages, and proposed system will work for other mixed languages as well. It would be interesting to analyse the results for different code-mixed languages. In future, researchers can also analyse the performance of all word embeddings, including fastText, GloVe, BERT and MuRIL. The proposed approach can be applied to any other NLP problem such as text summarization, fake news detection, Hate speech detection etc. having code-mixed dataset. In future, Type II fuzzy can also be the part of the correction system to handle the uncertainties of type 1 fuzzy, and it would be interesting to analyse the results.

Chapter 4

Word Sense Disambiguation

The term ‘apple’ means the fruit or the company depending upon the context of the sentence in which it is being used. Resolving the problem of semantic ambiguity in the words of a given sentence is known as Lexical semantic identification or Word Sense Disambiguation (WSD) [89]. This problem is crucial because it is often one of the intermediate steps in information retrieval, question answering, machine translation and many other natural language processing applications. The manual process of sense tagging is a tedious task; therefore, there is a need to automate this process. In this work, the terms WSD and LSI are used interchangeably.

Although the existing graph-based solutions eliminate the need for a labelled dataset, but they do not consider the fuzziness or degree of connectedness between the words in the graph. To overcome the mentioned problems, we propose a fuzzy graph-based approach. The core idea behind the proposed method is to use a fuzzy membership function to provide weight to the edges of the graph. In natural language, there is a degree to which the words are related to each other [5]. It is, therefore, possible to find the degree of association between the words. The higher the degree, the more related the words are to each other. This degree can be a mapping $W \times W \rightarrow [0,1]$, that is, by a fuzzy membership function (μ).

The novelty of the proposed work can be mentioned as this is the first work that exploits the idea of assigning weights to the edges of the fuzzy wordnet graph for Lexical Semantic Identification (LSI) using deep learning. The proposed work used deep learning-based word embedding i.e. Bidirectional Encoder Representations from Transformers (BERT) for LSI task for the first time. The correct sense is further identified using fuzzy measures of centrality.

4.1 Proposed Work for WSD

Fuzzy centrality measures [5, 6] (fuzzy degree, fuzzy betweenness and fuzzy PageRank) have been used in this work.

Fuzzy Degree

Degree centrality in an undirected graph is the number of edges connected to a given node. In a fuzzy graph, it is described as follows [5]

$$deg_f(i) = \sum_{u:(u,i)\in E} \mu_{ui} \tag{4.1}$$

In equation 4.1, μ_{ui} refers to the fuzzy weight of the edge between node *u* and node *i*.

Fuzzy Betweenness

Betweenness centrality refers to the fraction of the number of shortest paths that pass through a given node. Higher the betweenness, the more important that node is. In a fuzzy graph, it is given by [6]

$$between(i) = \frac{\rho_i(s,t)}{\rho(s,t)} \tag{4.2}$$

In equation 4.2, $\rho_i(s,t)$ depicts the number of shortest paths between nodes *s* and *t* that pass through the node *i*.

Fuzzy PageRank

Fuzzy PageRank takes into account the fact that not all links are equal. The number and quality of the links are used to find the relevance of a node [6]. In a fuzzy graph, it is given a

$$PR_f(i) = \frac{(1-d)}{|V|} + d \times \sum_{j:(j,i)\in E} \frac{\mu_{ji}}{\sum_{k:(j,k)\in E} \mu_{jk}} \times PR_f(j) \tag{4.3}$$

In equation 4.3, *d* refers to the damping factor [90] whose value is taken as 0.85, μ_{ji} is the fuzzy weight assigned to the edge between nodes *j* and *i*.

Following are the steps of the proposed algorithm:

1. Initially, sentence *S* is the input sentence. Next pre-processing is performed that includes stop words removal using NLTK, the rest of the words are POS-tagged, and only nouns and verbs are considered. These words are known as content words.
2. A fuzzy graph is then created using WordNet, which can be perceived as a graph. The *synsets* are the nodes, and the relations (e.g. *hyponym, holonym*) between them

are the edges. All the senses of the content words are taken as the initial set of vertices. DFS (Depth First Search) is then performed on WordNet (all relations are considered) starting from one of these initial vertices till another sense of this same set is reached. Intermediate vertices and edges of this path are then added to the set of vertices and edges of the graph.

3. Fuzzy weight is then provided to the edges using pre-trained BERT embeddings. BERT provides different embedding for each sense of a word. WordNet provides an example sentence for each sense which is used to calculate the embedding. Cosine similarity between embeddings of the two nodes making up an edge is calculated and used as the fuzzy edge weight.

Algorithm 4.1: Lexical Semantic Identification using *BERT embeddings*

```
Input: Sentence $S = (s_1, s_2, s_3 \ldots\ldots s_n)$ where $s_i$ are words in the sentence, $1 \le i \le n$

Step 1: Pre-process the sentence using the following steps
    i.   Remove stop words from $S$
    ii.  Remove words with pos tag other than noun and verb
             $S = (s_1, s_2, s_3 \ldots\ldots s_m)$
         where $s_i$ is the content word left after pre-processing
Step 2: Create fuzzy graph $F = (V,E)$ where the set of vertices is represented by $V$ the set of
        edges by $E$.
    i.   Initialise $V$ by adding all the senses of the content words from WordNet
             $V = \bigcup_{i=1}^{m}(senses)$
             $E = \phi$
    ii.  For each node $v$ in $V$ perform the following steps
        i.   Take $v$ as the starting node and perform DFS till level $l$ (l is the limiting
             value & is usually between 5 and 8) or till a vertex $v`$ is reached where $v` \subset V$
        ii.  Add all the vertices in the formed path to $V$ and edges between those to $E$
                 $V = V \cup (v, v_1, v_2, \ldots\ldots v`)$
                 $E = E \cup ((v, v_1), (v_1, v_2) \ldots\ldots (v_i, v`))$
        iii. Provide each edge in $E$ with a fuzzy weight using pre-trained BERT
             embeddings and examples of the two nodes from WordNet. Let $M_i$ and $M_j$
             be the word embeddings for node $i$ and node $j$ of the graph
                 $w_{ij} = cosine\_similarity(M_i, M_j)$
Step 3: Calculate the fuzzy measures of centrality using the equations 4.1 – 4.33
Step 4: For each content word the sense with the maximal value of a particular measure of
        centrality is taken as the correct sense
```

4. Fuzzy centrality measures are then used to calculate the correct senses. (Please refer algorithm 4.1)

For the Illustration of the proposed method, The sentence "*She is drinking milk*" is taken, and after pre-processing, we have left with "*drink*" and "*milk*" as the content words. The term "drink" has a total of eleven senses in WordNet, while "milk" has seven senses. These act as the initial vertices. DFS is then performed on the WordNet graph and the fuzzy weight for each edge is calculated using pre-trained BERT embeddings. Fig. 4.1 shows the formed graph for the example sentence.

Fuzzy measures of centrality are then used to obtain the correct senses. It can be seen from calculation that fuzzy degree and fuzzy PageRank have the highest value for "*drink.v.1*" and "*milk.n.1*". The values of fuzzy betweenness came out to be 0.0 for all the senses of "*milk*" and "*drink*". Highest fuzzy degree for *drink.v.1* is 1.420 and for *milk.n.1* is 0.374. Fuzzy PageRank for *drink.v.1* is 0.1 and for *milk.n.1* is 0.041.

According to WordNet, the definition of "*milk.n.1*" is "*a white nutritious liquid secreted by mammals and used as food by human beings*". Similarly, the definition of "*drink.v.1*" is "*take in liquids*". Both these definitions align with the intended meaning of the sentence.

4.2.Experimental Setup and Result Discussion

SemCor and Senseval-3 are the two datasets that are used for evaluation [29]. SemCor has been derived from the Brown corpus. It contains more than 2,00,000 content words which have been manually sensed tagged using WordNet sense. Senseval-3 [91] has been derived from the Penn Treebank corpus and consists of 2,018 content words annotated using WordNet senses. We compare our work with baseline models. The first baseline method, for each ambiguous content word, picks a sense randomly. Lesk's WSD model is another baseline method [27]. The next baseline method uses unweighted graphs and measures of centrality for disambiguating words [7]. Next is the comparison with the work, which is WSD using fuzzy semantic relations. The proposed work also compared

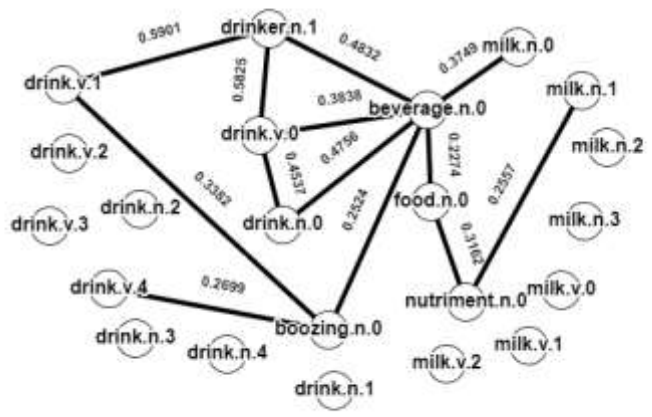


Fig. 4.1. Fuzzy Graph of Example Sentence

with the latest methods CEA [92] and SCSMM [93]. Both these approaches used crisp relations graph-based methods whereas the proposed method fuzzified the network for more close analysis of the real-world. The Key Player measure of centrality for this method as it gave the best results. It has been observed that the first sense heuristic, that is, the sense for an ambiguous word with the highest frequency in SemCor acts as an upper bound and unsupervised, as well as most supervised methods, as unable to outperform it.

Only words with more than one sense in WordNet are considered for disambiguation; that is, only polysemous words are considered. Table 4.1 depicts the precision (P), recall (R) and F-measure (F) values of the baseline models and the proposed method on the SemCor dataset. Table 1 depicts the results for Sensevel-3. Our method's performance is compared with the best semi-supervised and unsupervised systems for the same. Our method performs better than the best-unsupervised system (Sussex) and gives comparable results to the best semi-supervised method (IRST-DDD), which runs on an algorithm related to similarity. The fuzzy PageRank measure of centrality gives the best results as it takes into account this fuzzy weight.

Table 4.1: Results on SemCor and Senseval-3 Datasets

SemCor Dataset				Senseval-3 Dataset			
Method	**P**	**R**	**F**	**Method**	**P**	**R**	**F**
Random	23.51	23.51	24.0	Lesk WSD	44.05	32.91	37.67
Lesk WSD [	46.72	37.11	41.36	Unsupervised (Sussex)	47.80	41.26	44.28

Unweighted graph-based WSD (KPP)	51.01	38.26	43.7	Semi-supervised (IRST-DDD)	55.36	42.95	43.11
WSD Fuzzy semantic relations	51.08	40.24	45.01	WSD Fuzzy semantic relations	53.18	38.02	44.33
Proposed Fuzzy Degree	52.94	41.26	46.37	SCSMM	56.2	41.4	47.7
Proposed Fuzzy Betweenness	52.63	37.32	43.64	CEA [	54.3	37.3	44.2
Proposed Fuzzy PageRank	54.88	42.54	47.92	Proposed Fuzzy Degree	54.82	40.47	46.56
Proposed First Sense	**72.45**	**69.37**	**70.87**	Proposed Fuzzy Betweenness	52.73	39.68	45.28
				Proposed Fuzzy PageRank	**56.94**	**41.39**	**47.93**

4.3 Summary

Fuzzy graphs consider membership functions to provide weight to the edges. In the proposed method, we consider cosine similarity between vectors provided by BERT embeddings It is found that the proposed fuzzy graph-based method, where the fuzzy weight is provided using BERT embeddings, incorporates the context of the word. Fuzzy centrality measures are further utilized for disambiguating the content word. The proposed work has been evaluated using SemCor and Senseval-3 datasets. It provides superior results than all the baseline models using SemCor dataset and the best-unsupervised system for Senseval-3. The proposed work shows that fuzzy PagerRank outperforms the other fuzzy measures of centrality. One limitation of the proposed method is the dependency on the WordNet for graph generation. Although with the help of WordNet, we are able to incorporate semantics in network but still limited word sets and the absence of multi-words such as collocations, verbal phrases, idioms, etc. are its major constraints. In Future work, this limitation may be mitigated by incorporating other knowledge bases for instance conceptNet. In the future, various other fuzzy measures of centralities can be tested for disambiguation. Also, other algorithms for generating embeddings can be used. Other applications, such as graph-based keyword extraction, text summarization etc., can also be explored using the proposed algorithm.

Chapter 5

Automatic Construction of Interval-Valued Fuzzy Hindi WordNet

Human mind is trained from childhood to handle the language complexities. But for computers to comprehend multiple meanings and context precisely is an arduous process. To make the language understandable by computers, it is mandatory to comprehend the words and their association with other words. To represent this semantic association, researchers created lexicons. WordNet [94] is a lexicon that is available for almost all languages and extensively being used for many natural language processing applications. WordNet for Hindi, called *Hindi WordNet*, having 16 relations between words, is developed by the Indian Institute of Bombay (IIT-B) in early 2000 [95].

The Hindi WordNet (HWN) is an available online system that comprises various semantic and lexical relations between the Hindi words. It has been explored for almost every Hindi language processing application, such as Word processing Information Retrieval (IR), Question Answering systems (QRs), Machine Translation (MT), etc.. HWN also has been used as a base resource to develop new lexicons such as Hindi SentiWordNet [96]. Hindi WordNet has 1.05,423 unique words and 40465 synsets with various crisp relations between the words[1]. However, in real-life situations, relationships between words (as used to convey natural language) are not crisp; rather, they are a matter of degree, with a gradual transition from being related to not being related. For instance, as shown in Fig. 5.1 (a), there is a "meronymy" semantic relation between फल and गुठली. In classical Hindi WordNet, all relations are crisp, but it is easy to observe that while some fruits (such as bananas) don't have गुठली, the majority of them have. Words in natural language have partial relations, which crisp classical Hindi WordNet does not handle. Fig. 5.1(b) shows fuzzified relations of Hindi WordNet.

In this direction, Jain & Lobiyal investigated this concern and proposed Fuzzy Hindi WordNet (Type-1 Fuzzy) that comprises fuzzified relations [5]. This was the first work to incorporate real-life uncertainties in Hindi WordNet. The authors proved the better performance of fuzzy Hindi WordNet over classical Hindi WordNet by handling word sense disambiguation. Jain et al. proposed a very compelling idea of Fuzzy Hindi WordNet (FHWN), and they assigned the link strengths between two words in the FHWN network by linguistic experts. However, the concept of involving experts is impractical when we want to incorporate a large number of the relation between words. For instance, Hindi WordNet has 1,05,423 unique words/nodes, and the possible relation between these unique words will be ${}^{n}C_{2}$ = 5556951753 relations, where n is the unique words. It is next to impossible to assign membership value to this large number of relations with the help of experts. Their idea is very significant to express the relations between words according to real-world scenarios, but it is virtual until researchers can use it for NLP applications. In this direction, it is desirable to generate fuzzy Hindi WordNet relations automatically. One more issue in the literature is that the authors considered average of the fuzzy membership values taken from experts and converted into a single value, which results in loss of opinion and leads to a loss in the valuable data. This is again not handled by existing work.

The proposed work overcomes the shortcomings as mentioned above in the following ways:

1. Overcome the requirement of enormous human resources (linguistic experts) for membership value assignment: The work proposed Interval- Valued Fuzzy Hindi WordNet (IVFHWN) that automatically identifies Interval- Valued Fuzzy semantic relations between Hindi words using lexico-syntactic patterns and various Word embeddings. We render IVFHW as Interval- Valued graph where nodes/vertices represent the Hindi words and edges represent the IVF relations between words. Fig. 5.1 (b) shows the IVFHW relations for the word 'फल'. The words "फल and पेड" are connected to each other with IVF holonymy relation having [0.66, 0.75] membership value. Lexico-syntactic patterns are generalised linguistics structures that used to extract semantic relationships between words.

2. Handles the uncertainty in the Type-1 fuzzy membership degree. In this chapter, we have incorporated Interval- Valued Fuzzy relations, a generalised form of Type-2 Fuzzy relations. The motivation behind using IVFs is uncertainty in the membership degree of

Type-1Fuzzy. This part also resolved the issue of loss of information occurs in averaging values taken by different experts.

To evaluate the proposed Interval- Valued Fuzzy Hindi WordNet, we used a heath domain corpus publicly available at the IIT Bombay website. In order to assess the proposed method, the mean square error (MSE) is calculated between the expert value and the proposed value. To arrive at the expected value, we enlisted the help of a group of famous lexicographers (linguists who specialise in Hindi) and fuzzy logic experts. The results demonstrated proposed method (IVFHWN) is providing good results.

After the introduction Section 5.1 deals with the proposed work, which consists of the definitions of Interval-Valued Fuzzy Hindi WordNet and relations with their identification and membership degree assignment methods. Section 5.2 describes the experimental setup, discussion of the results and comparison of the available systems. Finally, in Section 5.3, has the summary of chapter.

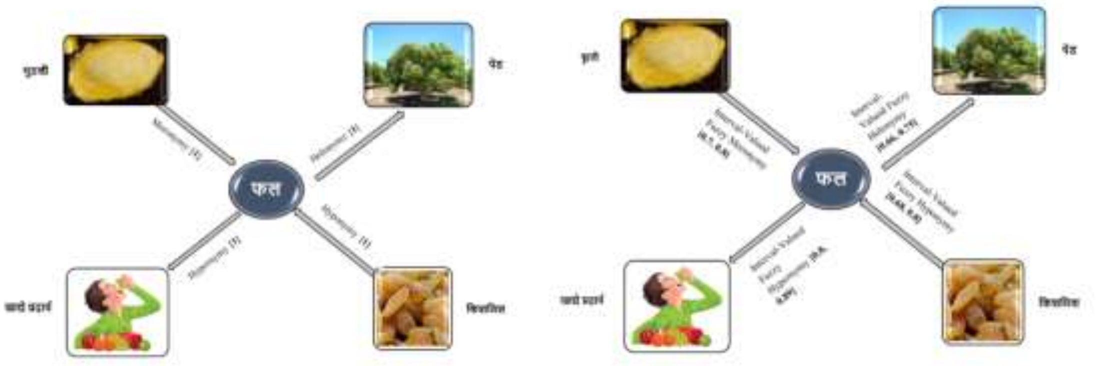


Fig. 5.1 (a) Crisp relations for the word 'फल' in Hindi WordNet   Fig. 5.1(b) Interval-valued Fuzzy relations for the word 'फल' in proposed IVFHW

5.1. Proposed Interval-Valued Fuzzy Hindi WordNet (IVFHW)

In the following section, we define Interval-Valued Fuzzy Hindi WordNet (IVFHWN) and various IVF relationships. This section also comprises the identification of IVF relationships and automatic assignment of membership value to the IVF relationships. These relationships are identified using various lexico-syntactic patterns and If-Then rules, whereas different word embeddings are used to assign membership values automatically.

5.1.1 Definition of Interval-Valued Fuzzy Hindi WordNet and Its relations

Definition A. An Interval-Valued Fuzzy Hindi WordNet Graph is denoted as IVFHWN (σ, $\mu_{\tilde{r}}$), which is a pair of functions σ : W→{1} and μ_r : W×W→ $[0 \leq \mu_{\tilde{r}_L} \leq \mu_{\tilde{r}_U} \leq 1]$, where σ is a set of nodes representing the set of Hindi open class words (set of synsets) with the unity strength of every node. It means every Hindi word entirely belongs to Interval- Valued Fuzzy Hindi WordNet. $\mu_{\tilde{r}}$, which is Interval- Valued Fuzzy membership degree denotes the strength of an edge between two Hindi words, where $\mu_{\tilde{r}_L}$ *and* $\mu_{\tilde{r}_U}$ Indicates the lower and upper membership value of relation r̃, respectively.

For example, Fig. 5.2 shows an Interval- Valued Fuzzy Hindi WordNet graph where w_1, w_2, w_3…… w_5 are the Hindi words; word w_1 is linked with word w_2 with hypernymy ($\widetilde{hr}$) relationship having IV fuzzy degree of strength $\mu_{\widetilde{hr}}$ ϵ [0.2,0.4]; similarly, word w_2 is connected with word w_3 with Meronymy (M̃) relationship having IV fuzzy degree of strength $\mu_{\tilde{M}}$ ϵ [0.3,0.4] and so on.

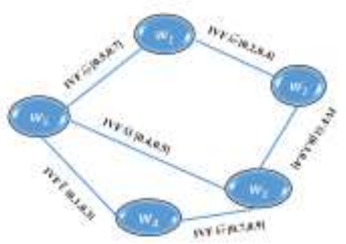


Fig. 5.2.: Interval-Valued Fuzzy Graph

In the Interval-Valued Fuzzy WordNet network, there are sixteen fuzzy relationships (same part-of-speech and cross part-of-speech) between words, and they are defined formally as follows:

Definition B: Interval-Valued Fuzzy Association (Ã*s*) relationship represented as (w_1, w_2, $\mu_{\tilde{A}s}$) and defined as IVF relation between w_1 and w_2 (two synsets) where $w_1, w_2 \in W$ and $\mu_{\tilde{A}s}$: $W \times W \rightarrow [\mu_{\tilde{A}s_L}(x), \mu_{\tilde{A}s_U}(x)]$, such that w_1 and w_2 are approximately synonymy where W denotes to the set of all synsets, $\mu_{Ãs}$ indicating the degree of strength of the relationship with $[\mu_{Ãs_L}(x), \mu_{Ãs_U}(x) \in [0,1]$ indicating the lower and upper interval of $\mu_{Ãs}$. IVF association (Ãs) represents the partial similar meaning between two words, for instance (सोफा, कुर्सी, [0.6,0.75]) where interval [0.6,0.75] shows the strength of the relationship in the form of interval. More examples are: (मतदान, राय, [0.4,0.6]) (दस्ता, कॉपी, [0.54, 0.72]) (मेज, स्टूल, [0.5,0.7])

Definition C: Interval-Valued Fuzzy Hypernymy ($\widetilde{hr}$) and Hyponymy ($\widetilde{hp}$) can be represented as (w_1, w_2, $\mu_{\widetilde{hr}}$ / $\mu_{\widetilde{hp}}$) and defined as IVF relation between w_1 and w_2 (two synsets) where $w_1, w_2 \in W$ and $\mu_{\widetilde{hr}}$: $W\times W \rightarrow [\mu_{\widetilde{hr}_L}(x), \mu_{\widetilde{hr}_U}(x)]$, $\mu_{\widetilde{hp}}$: $W\times W \rightarrow [\mu_{\widetilde{hp}_L}(x), \mu_{\widetilde{hp}_U}(x)]$, such that w_1 is an approximate subset/superset of w_2 where W denotes the set of all synsets, $\mu_{\widetilde{hr}}$ / $\mu_{\widetilde{hp}}$ indicating the degree of strength of the relationship with $\mu_{\widetilde{hr}_L}$(x), $\mu_{\widetilde{hr}_U}$(x), $\mu_{\widetilde{hp}_L}$(x), $\mu_{\widetilde{hp}_U}$(x) indicating the lower and upper interval value of $\mu_{\widetilde{hr}}$ / $\mu_{\widetilde{hp}}$.
For instance, there is IVF hypernymy/hyponymy relation between (नेल कटर and हथियार with interval membership value [0.6, 0.8] where 0.6 is the lower interval value and 0.8 is the upper interval value. Similarly, (बईमान and दुर्जन are related with strength of [0.4, 0.5]).

Definition D: Interval-Valued Fuzzy Meronymy (M̃) and Holonymy (H̃o) are semantic relations between two synsets w_1 and w_2 and used to capture the part-whole relationships between w_1, w_2, where $w_1, w_2 \in W$ and $\mu_{\tilde{M}}$: $W\times W \rightarrow [\mu_{\tilde{M}_L}(x), \mu_{\tilde{M}_U}(x)]$, $\mu_{\tilde{H}o}$: $W\times W \rightarrow [\mu_{\tilde{H}o_L}(x), \mu_{\tilde{H}o_U}(x)]$, where W denotes the set of all synsets, $\mu_{\tilde{M}}$ / $\mu_{\tilde{H}o}$ indicating the degree of strength of the relationship. For example, (बटन, कुर्ता [0.8,0.9]), (फूल, पेड़ [0.65, 0.74]), (बिजली बारिश [0.5, 0.66])

Definition E. Interval-Valued Fuzzy Antonyms (Ã): It is a lexical relation between two words with approximately opposite meanings in some context. Interval- Valued Fuzzy Antonyms can be represented as (w_1, w_2, $\mu_{\tilde{A}}$) where $\mu_{\tilde{A}}$: $W\times W \rightarrow [\mu_{\tilde{A}_L}(x), \mu_{\tilde{A}_U}(x)]$ and

w_1, $w_2 \in W$. Illustration: (बेटी, लड़का [0.6, 0.78]), (शाम, दिन [0.7, 0.8]) and (मुर्दा बेहोश [0.6, 0.73])

Definition F. Interval-Valued Fuzzy Entailment (Ẽ): It is a one-way semantic relation that holds between two synsets of verbs if the truth of verb *y* partially follows the truth of verb *x*. It can be repressed as $v_1 \rightarrow v_2$ having IVF membership $\mu_{\tilde{E}}: V \times V \rightarrow [\mu_{\tilde{E}_L}(x),\ \mu_{\tilde{E}_U}(x)]$, v1, v2 ∈ V and V (V⊂W) is the set of all verbs in the Hindi language.
For instance, (सब्जी काटना, खाना बनाना [0.5, 0.62]) , (निरोगी होना, व्यायाम करना [0.8, 0.9]), (बुरी आदते होना, बुरे संस्कार होना [0.4,0.54])

Definition G. Interval-Valued Fuzzy Troponymy (T̃): This relation relates two synsets of verbs, verb v_A denoting elaboration of verb v_B in an approximately specific manner. Interval- Valued Fuzzy Troponymy can be represented as (v_A, v_B, $\mu_{\tilde{T}}$), where $v_A, v_B \in V$ and $V \subset W$ is the set of verbs in Hindi Language. and $\mu_{\tilde{T}}: V \times V \rightarrow [\mu_{\tilde{T}_L}(x),\ \mu_{\tilde{T}_U}(x)]$ is the Interval- Valued membership value for fuzzy tropynymy. Illustration: (भागना, चलना [0.6, 0.69]), (हसना, खुश होना [0.8, 0.9]) and (सिलाई करना, सीखना [0.5,0.63])

Definition H. Interval-Valued Fuzzy Causative (C̃) is a lexical relation which associates the causative verbs and shows interdependencies between them. For this relation, the membership value is always unity because it shows the relation between the different morphological forms of a verb. For Illustration **(**पीना पिलाना, 1) and (सोना सुलाना , 1)

Definition I: Interval-Valued Fuzzy Ability Link ($\tilde{A}l$) is semantic relation, represented as (*n. v*, $\mu_{\tilde{A}l}$), where $n \in N$, $v \in V$ and N, $V \subset W$ is the set of nouns and verbs in Hindi Language respectively. $\mu_{\tilde{A}l}$ is the Interval- Valued membership value for the fuzzy ability link. This relation specifies the inherited feature of a nominal word. For Instance (पंछी, उड़ना [0.7, 0.8])

Definition J: Interval- Valued Fuzzy Capability Link ($\tilde{C}_L$) is semantic relation, represented as (*n. v*, $\mu_{\tilde{C}_L}$), where $n \in N$, $v \in V$ and N, $V \subset W$ is the set of nouns and verbs in Hindi Language respectively, and $\mu_{\tilde{C}_L}$ is the Interval-Valued membership value for fuzzy capability link. This relation specifies the developed feature of a nominal word. For instance, the capability of नौकरी करना of a आदमी is more than the capability of नौकरी करना of a लड़का because in most of the cases आदमी refers to an adult whereas लड़का refers to a juvenile. This relation can be represented as: (आदमी, नौकरी करना [0.9, 0.98]) (लड़का, नौकरी करना[0.5,0.6])

Definition K: Interval- Valued Fuzzy Function Link ($\tilde{F}_L$) is represented as ($n.\ v,\ \mu_{\tilde{F}_L}$), where $n \in N$, $v \in V$and $N,\ V \subset W$ is the set of nouns and verbs in Hindi Language respectively, and $\mu_{\tilde{F}_L}$ is the Interval- Valued membership value for fuzzy function link. It is a semantic relation that specifies the function of a nominal word. This relation can be understood by an example like: "पढ़ाना is a function of a अध्यापक ". In an ideal world, all teachers would be expected to teach; nevertheless, many administrative professors (such as the Vice-Chancellor of a university) are not allocated any teaching responsibilities. In addition, some teachers are given a heavier teaching load than others. A professor, for example, is given a lower teaching load than an assistant professor or associate professor. Professors, on the other hand, are given more administrative and research responsibilities (as well as a lower teaching load) than assistant and associate professors. The strength of the respective function is used to determine the weight of a fuzzy function connection.

Definition L: Interval- Valued Fuzzy Attribute ($\widetilde{Att}$) is a semantic relationship between the synset of a noun and adjective. It represents the partial properties of a noun. It can be represented as ($n.\ a_d$, $\mu_{\widetilde{Att}}$), where $n \in N$, $a_d \in A_d$ and N, $A_d \subset W$ is the set of nouns and adjectives in Hindi Language, respectively, and $\mu_{\widetilde{Att}}$ is the Interval- Valued membership value with upper and lower intervals. For instances (सुशील ,लड़की [0.5, 0.6]), (बंजर, भूमि [0.7,0.78])

Definition M: Interval- Valued Fuzzy Modifies Noun ($\tilde{M}_N$) is the semantic relationship between adjectives and nouns. It comprises cases when adjectives can only go with specific nouns. For this relation, the membership value is always unity. It can be represented as ($n.\ a_d$, $\mu_{\tilde{M}_N}$), where $n \in N$, $a_d \in A_d$ and $N, A_d \subset W$ is the set of nouns and adjectives in Hindi Language respectively, and $\mu_{\tilde{M}_N} = \{1\}$ is the Interval- Valued membership value. For example (संस्कारी, इंसान , 1)

Definition N: Interval- Valued Fuzzy Modifies Verb ($\tilde{M}_v$) is the semantic relation between verb and adverb such that an adverb modifies the specific verb. For this relation, the membership value is always unity. It means this is always a classic relation. This relation is anti-symmetric, anti-reflexive, and max-*-nontransitive. For instance (तेज ,चलना, 1)

Definition O: Interval- Valued Fuzzy Derived From ($\tilde{D}_F$) specifies the origin (root) form from which a word is derived. It is a lexical relation, and the membership value for this relation is always unity. This relation is anti-symmetric, anti-reflexive, and max-*-nontransitive. For instance (सुंदरता, सुंदरी, सुंदर , 1)

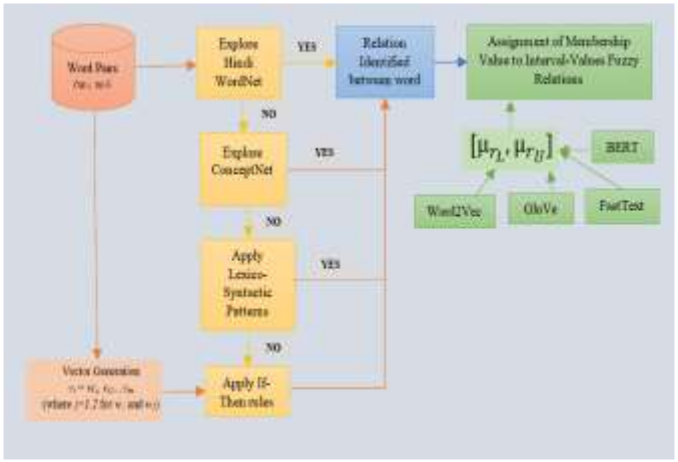


Fig. 5.3. Work Flow Diagram of the Proposed IVFHW

5.1.2. Identification of Interval-Valued Fuzzy relations

This section discusses the lexico-syntactic patterns and If-then rules for the identification of IVF relationships between words. All the word pairs are first explored in Hindi WordNet; if the relation is found in WordNet, it goes to the next step: the assignment of membership value, or else word pairs are searched in conceptNet. It not available in conceptNet also, then below discussed lexico-syntactic patterns and if-then rules are used to find the relationship. The overall architecture is shown in Fig. 5.3.

1. Identification of IVF association relation includes the following steps:
   a) Let say w_1 and w_2 represent words, and the objective is to identify IVF $\tilde{A}s$ the relation between them, for this, the first step is to generate vector $v_i = vi_1, v_{i2}...v_{im}$ (where $i=1,2$ for w_1 and w_2). Vector v_1 and vector v_2 comprise co-occurred *VP* and *NP* and *AdjP*. For instance, vectors for words (स्कूटर and मोटरसाइकल) are shown in fig.5.4
   b) The next step is calculating the weight of each term in a vector using normalize Tf×idf (Term frequency multiplied by inverse document frequency) method shown in formula.

$$\text{Tf} \times \text{idf}\,(t, d_i) = \frac{(0.5+0.5\,\frac{tf_{it}}{Max_{j=1,2,3...,L}tf_{ik}})log\frac{N}{df_t}}{Max_{j=1,2,3...,L}\{(0.5+0.5\,\frac{tf_{it}}{Max_{j=1,2,3...,L}tf_{ik}})log\frac{N}{df_j}\}}$$

Where, $tf_{it}$is the frequency of term *t* appearing in document d_i, df_t is the total number of documents containing the term *t* . L is the number of terms contained by d_i, and *N* is the number of documents in the corpus.

If-Then rule 1: If vector v_i and vector v_j include almost identical words and the weights of the words in both vectors are equivalent, then these two vectors should be similar and have an IVF association relationship. From Fig. 5.4, it can illustrate both (स्कूटर, मोटरसाइकल) has almost the same terms with equivalent weights (दो पहिया , ब्रेक, तेज, गति, धातु). So, there is IVF $\tilde{A}s$ relationship with membership degree interval (स्कूटर, मोटरसाइकल [0.8, 0.9]).

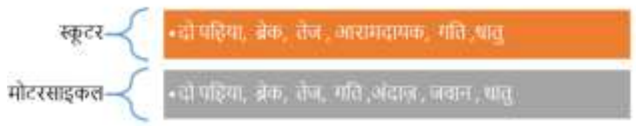

स्कूटर
•दो पहिया, ब्रेक, तेज, आरामदायक, गति,धातु
मोटरसाइकल
•दो पहिया, ब्रेक, तेज, गति ,अंदाज़, जवान ,धातु

Fig. 5.4. Vector for words (स्कूटर, मोटरसाइकल)

2. Identification of IVF Hypernymy ($\widetilde{hr}$) and Hyponymy ($\widetilde{hp}$) relation is done using two methods 1) indicative lexico-syntactic patterns, i.e. particular pattern indicates a semantic relation between terms. In literature various authors took an interest in the automatic acquisition of lexical relations with the goal of building up large lexicons for natural language processing [53,54]. In the section, we also discovered a set of lexico-syntactic patterns to identify IVF hypernymy/hyponymy relation from the text. Below is the list of discovered lexico-syntactic rules, followed by illustrative sentences.

Rule 1: NP_0 जैसे {NP_1, NP_2,….,(and/or)} NP_n}

such that it implies

For all NP_i, $1\leq i \leq n$, IVF hyponymy (NP_i, NP_0)

For instance: *S1*: कई प्रकार के मिठाई जैसे शकरपारे, आइस्क्रीम, गुड ,Thus from *S1* we conclude: IVF hyponym ("शकरपारे", "मिठाई" [0.6,0.7]), IVF hyponym ("आइस्क्रीम", "मिठाई" [0.7, 0.8]), IVF hyponym ("गुड", "मिठाई" [0.3, 0.4])

Rule 2:NP_0{,NP_i}*और NP_n NP_{n+1}

such that it implies

For all NP_i, $1\leq i \leq n-1$, IVF hyponymy (NP_i, NP_{n+1}), IVF hyponymy (NP_0, NP_{n+1}), IVF hyponymy (NP_n, NP_{n+1})

For instance: *S2*: कैंची और नेलकटर हथियार है

Thus from *S2* we conclude: IVF hyponym ("कैंची", "हथियार" [0.4,0.5]), IVF hyponym ("नेलकटर ", "हथियार"[0.3, 0.4).

Rule 3: NP_0 एक NP_1 है

such that it implies IVF hyponym (NP_1, NP_0)

For instance: *S3*: किशमिश एक फल है

Thus from *S3*, we conclude: IVF hyponym ("किशमिश", "फल" [0.7,0.8])

Rule 4: NP_0{,NP_i}* (और/ या) बाकी NP_{n+1}

such that it implies For all NP_i, $1\leq i \leq n-1$, IVF hyponymy (NP_i, NP_{n+1}), IVF hyponymy (NP_0, NP_{n+1}), IVF hyponymy (NP_n, NP_{n+1})

For instance: *S4*:, नील, खरोच, जलना ओर बाकी घाव .....

Thus from *S4* we conclude: IVF hyponym (“नील”, “घाव” [06, 0.7]), IVF hyponym (“खरोच”, “घाव” [0.5,0.6]), IVF hyponym (“जलना”, “घाव”[0.8,0.9])

For instance: *S5*: मीनार, गुफा, बॅवोली या बाकी एतिहसिक स्मारक....

Thus from *S5* we conclude: IVF hyponym (“मीनार”, “एतिहसिक स्मारक”[0.8,0.9]), IVF hyponym (“गुफा”, “एतिहसिक स्मारक”[0.6, 0.7]), IVF hyponym (“बॅवोली”, “एतिहसिक स्मारक” [0.7, 0.82])\

Rule 5: NP_0 विशेष रूप से {NP_1, NP_2,….,(and/or)} NP_n}

such that it implies

For all NP_i, $1 \leq i \leq n$, IVF hyponymy (NP_i, NP_0)

For instance: *S6*: कॉलेज कर्मचारी विशेष रूप से प्रोफेसर, अनुसंधान साथी

Thus from *S6* we conclude: IVF hyponym (“कॉलेज कर्मचारी”, “प्रोफेसर”), IVF hyponym (“कॉलेज कर्मचारी”, “अनुसंधान साथी”),

Rule 6: $NP_0\{,NP_i\}^*$ सहित NP_{n+1}

such that it implies For all NP_i, $1 \leq i \leq n-1$, IVF hyponymy (NP_i, NP_{n+1}), IVF hyponymy (NP_0, NP_{n+1}), IVF hyponymy (NP_n, NP_{n+1})

For instance: *S7*: चाकू, सुई, नई कटर सहित सभी हथियार

Thus from *S7* we conclude: IVF hyponym (“हथियार”, “नई कटर”), IVF hyponym (“हथियार”, “सुई”)

2) Creating vectors (v_i, v_j) of terms as discussed above and assigning them weight using tf-idf,

If-then rule 2: if $v_i \subset v_j$ and $W_t(v_i) > W_t(v_j)$, then vector v_i is said to dominate vector v_j and should be more specific than vector v_j

If one vector v_i is a subset of another vector v_j and the weight of terms in v_i vector is larger than those in vector v_j then there is IVF hyponymy/ hypernymy relation exists. An instance is shown in Fig. 5.5.

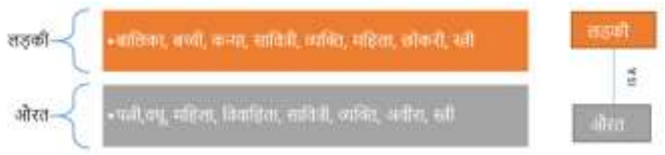

लड़की
•बालिका, बच्ची, कन्या, सावित्री, व्यक्ति, महिला, छोकरी, स्त्री
औरत
•पत्नी,वधू, महिला, विवाहिता, सावित्री, व्यक्ति, [illegible], स्त्री
लड़की
ISA
औरत

Fig. 5.5. Vector for words (लड़की, औरत)

3. Identification of IVF Meronymy (M̃) and Holonymy (H̃*o*) Relation:

1) identified using lexico-syntactic text patterns, i.e. particular pattern indicates a semantic relation between terms.

Rule 1:NP_0 मे NP_1 such that it implies IVF meronymy (NP_1, NP_0)

For instance: *S8*: लॅपटॉप मे बॅटरी होती है , Thus from *S8* we conclude: IVF meronymy (“बॅटरी”, “लॅपटॉप” [0.8,0.9])

Rule 2: NP_0, NP_1 हिस्सा such that it implies IVF meronymy (NP_0, NP_1)

For instance: *S9*: बाजू कमीज़ का हिस्सा है , Thus from *S9* we conclude: IVF meronymy (“बाजू”, “कमीज़” [0.6,0.7])

Rule 3: NP_0, NP_1 (ओर NP_2)* शामिल है such that it implies IVF meronymy (NP_1, NP_0) and IVF meronymy (NP_2, NP_0)

For instance: *S10*: टीम मे खिलाड़ी ओर सहायक शामिल है, Thus from *S10* we conclude: IVF meronymy (“खिलाड़ी”, “टीम”) and IVF meronymy (“सहायक”, “टीम” [0.5, 0.7])

Rule 4: NP_0, NP_1 दिखाई देते है such that it implies IVF meronymy (NP_0, NP_1)

For instance: *S11*: तारे आसमान मे दिखाई देते है , Thus from *S11* we conclude: IVF meronymy (“तारे”, “आसमान” [0.9, 0.98])

Rule 5: NP_0 पर, NP_1 है such that it implies IVF meronymy (NP_1, NP_0) For instance: *S12*: कॅमरा पर धूल है

, Thus from *S12,* we conclude: IVF meronymy (“धूल”, “कॅमरा” [0.6, 0.7])

Rule 6: NP_0, NP_1 पर उपलब्ध है such that it implies IVF meronymy (NP_0, NP_1)

For instance: *S13*: ग्लोकोज़ दुकान पर उपलब्ध है, thus from *S13* we conclude: IVF meronymy (“ग्लोकोज़”, “दुकान” [0.6, 0.8])

Rule 7: NP_0, NP_1 पाया such that it implies IVF meronymy (NP_1, NP_0)

For instance: *S14*: ग्रह पर पानी पाया, thus from *S14* we conclude: IVF meronymy (“पानी”, “ग्रह” [0.5, 0.6])

2) Creating vectors above (v_i, v_j) of terms as discussed above and assigning them weight using tf-idf (An instance shown in Fig. 5.6).

If-then rule 3: If $v_i \cap v_k = \varnothing$ but $v_i \cup v_k$ is a partition of another vector v_j then v_i and v_j are in a part-of relationship with v_k .

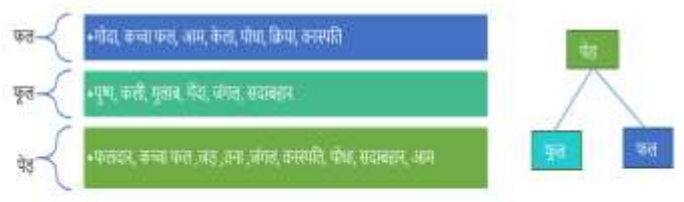

फल
•गोदा, कच्चा फल, आम, केला, पौधा, क्रिया, वनस्पति
फूल
•पुष्प, कली, गुलाब, पेड़, जंगल, सदाबहार
पेड़
•फलदार, कच्चा फल ,जड़ ,तना ,जंगल, वनस्पति, पौधा, सदाबहार, आम
पेड़
फूल
फल

Fig. 5.6. Vector for words (फल, फूल, पेड़)

4. Identification of IVF antonyms (Ã) Relation:

Identification of IVF antonymy relation depends on fuzzy association relation and crisp antonymy relation.

If-Then rule : If ($\mu_{\tilde{A}s}$ $(a, b)= [x, y]$ where $0 \leq x, y \leq 1) \cap$ (μ_A $(b, c)= 1)$ then ($\mu_{\tilde{A}}$ $(a, c)= [x, y]$ where $0 \leq x, y \leq 1)$. An instance for (औरत, लड़का) is shown in Fig. 5.7. Where, *a* is “औरत”, *b* is “लड़की” and *c* is “लड़का” and $\mu_{\tilde{A}}$ is IVF antonymy, A is crisp antonymy and $\mu_{\tilde{A}s}$ is IVF association relation

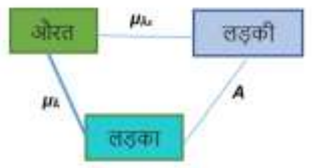


Fig. 5.7. An instance for Identification of IVF antonymy relation

5. Identification of IVF entailment Relation: To identify IVF entailment relationships, ConceptNet is Used. The relations "HasSubvent", "HasFirstSubevent", "HasLastSubevent", "HasPrerequisite" helped to identify entailment. Their definition with example is given in the Fig. 5.8.

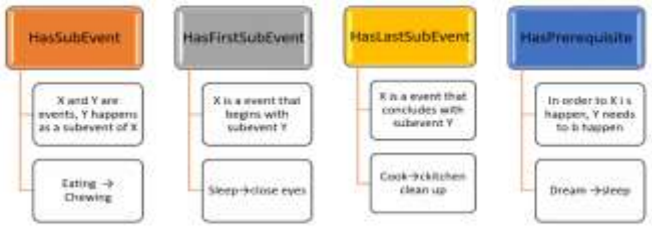

HasSubEvent
X and Y are events, Y happens as a subevent of X
Eating → Chewing
HasFirstSubEvent
X is a event that begins with subevent Y
Sleep→close eyes
HasLastSubEvent
X is a event that concludes with subevent Y
Cook→ckitchen clean up
HasPrerequisite
In order to X is happen, Y needs to b happen
Dream →sleep

Fig. 5.8: ConceptNet Relations for Entailment Identification

Another way is to identify is again Lexico-Syntactic Patterns like:

Rule 1: *VP*$_0$ मे VP$_1$ such that it implies IVF meronymy (*VP*$_1$, *VP*$_0$)

For instance: *S8*: खाना बनाने मे गर्मी लगती है, thus from *S8* we conclude: IVF meronymy ("खाना बनाना", "गर्मी लगना")

Rule 2: *VP*$_0$ से VP$_1$ such that it implies IVF meronymy (*VP*$_1$, *VP*$_0$)

For instance: *S10*: मेहनत करने से फल मिलता है, Thus from *S10* we conclude: IVF meronymy ("मेहनत करना", "फल मिलना"). *S11*: व्यायाम करने से स्वस्थ रहते है, Thus from *S11* we conclude: IVF meronymy ("व्यायाम करना", "स्वस्थ रहना")

Rule 3: *VP*$_0$ के बाद *VP*$_1$ such that it implies IVF meronymy (*VP*$_1$, *VP*$_0$)

For instance: *S12*: सोने के बाद खराटे आते है, Thus from *S12* we conclude: IVF meronymy ("सोना", "खराटे आना").

6. Identification of IVF Troponymy Relation: IVF troponymy relation exists between two verbs while IVF hypernymy/hyponymy exists between two verbs. So lexico-syntactic patterns and If-Then rules used for IVF hypernymy/hyponymy relation can be used for identification of IVF troponymy. To identify IVF troponymy relationships, ConceptNet is

used. The relations “MannerOf”, helped to identify entailment. Its definition, with the example, is given in Fig. 5.9.

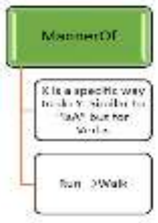

MannerOf
X is a specific way to do Y, similar to "IsA" but for Verbs.
Run ->Walk

Fig.5.9. ConceptNet Relation for Troponymy Identification

7. Identification of IVF Ability, Capability and Function Link Relation: This relation is learned from the data. All the NP+VP can be extracted for this relationship

8. Identification of IVF Attribute Relation: This relation is learned from the data. All the NP+Adj can be extracted for this relationship.

9. Identification of Causative, Modifies Noun, Modifies Verb, Derived Form Relation: These relations are always crisp relations with membership value unity. So, there is no need to identify them; specifically, these relationships can be extracted from Hindi WordNet directly.

5.1.3. Automatic Assignment of Membership Degree to Interval-Valued Fuzzy Relations

In literature, the multiple experts' opinion has been used to assign membership value to relations between two words/nodes in fuzzy graph. But it is a tedious and challenging task to determine the exact membership value by experts between millions of vertices/words. In this section, the automatic assignment of membership value to edges of a fuzzy graph is done by using word embeddings. In the proposed method, we used various word embeddings, including Word2Vec, Glove, Fasttext and BERT to find the belongingness between two words in a fuzzy graph. Assignment of Association, Hypernymy, Hyponymy, Antonymy, Meronymy, Holonymy, Entailment, Troponymy, Ability, Capability, Function Attribute relations are calculated using embeddings. And causative, Modifies Noun, Modifies Verb, Derived Form relation are always a crisp relation with membership value unity. For the dataset, we crawled a huge monolingual Hindi corpus from Hindi Wikipedia of size 7.2 GB. Words with a frequency of less than 2 in the total corpus are classified as unknown (out-of-vocabulary) terms. After preprocessing word2vec embeddings are calculated using the cosine similarity formula as follows:

Similarity score: It is used to denote the belongingness between the two nodes, and it's value lies between 0 and 1. The angle between two vectors is employed as a measure of divergence between the vectors, and the cosine of this angle is used to calculate cosine similarity. It is calculated using Equation (5.1) [40].

$$similarity(i, j) = \frac{i.j}{|i| \times |j|} \tag{5.1}$$

where, *i.j* is the dot product of the two vectors and is given by Equation (5.2) and |*i*|, |*j*| represents the magnitude of the two vectors and is given by Equation (5.3a)-(5.3b) and *similarity(i,j)* represents the cosine similarity between the two vectors, where n is the size of the vector.

$$i.j = \sum_{k=0}^{n} i_k \times j_k \tag{5.2}$$

$$|i| = \sqrt{\sum_{k=0}^{n} i_k^2} \tag{5.3a}$$

$$|j| = \sqrt{\sum_{k=0}^{n} j_k^2} \tag{5.3b}$$

Next to build FastText embeddings , the gensim library was used for implementation of FastText with dimensions of 50, 100, 200, and 300 (skip-gram architecture) [14]. once again. The default settings of gensim are utilised for the remaining parameters. FastText's official repository contains pre-trained word embeddings for the Hindi language. For GloVe embeddings [97] of 50, 100, 200, and 300 dimensions are created. A symmetric window of size 15 is used to build the co-occurrence matrix.

We train 300-dimensional BERT (Bidirectional Encoder Representations from Transformers) embeddings [98]. We employ sentence component embeddings, which were trained on a corpus with a vocabulary of 25000 words. This model was pre-trained using three Tesla K80 GPUs, each with 12 GB of memory. The official BERT repository has a multilingual model with 102 languages, including Hindi.

Let values provided by these four embeddings are: W_{2v}, G_L, F_T, and B

$$\mu_{\tilde{r}} = [min\ (W_{2v}, G_L, F_T, B_E), max\ (W_{2v}, G_L, F_T, B_E)] \tag{5.4}$$

where $\mu_{\tilde{r}}$ is the IVF membership value of the relation *r*.

5.2. Experimental Setup and Result Discussion

The proposed Interval- Valued Fuzzy Hindi WordNet is evaluated using a heath domain corpus, publicly available at the IIT Bombay website. Health corpus is part-of-speech tagged and comprises 90 files with approximately one hundred sentences in each file. In the corpus, the average number of open-class terms in a sentence is $8.85 \approx 9$. The number of nouns, Verbs and adjectives required for different IVF relations (mention in section) is 52230, 24291 and 22699, respectively.

Table 5.1 shows some of the sentences from the Hindi corpus, as well as the fuzzy relationships between the terms in the sentences. The table also mentions membership value taken from linguistic experts and membership value computed by the proposed method. The mean square error (MSE) between the expert value and the proposed value is calculated to evaluate the proposed approach.

MSE calculates that how much error the method has made during prediction. The MSE between the EV (Expert Value) and the value calculated by the proposed method (PM) can be calculated as:

$$MSE\ (EV, PM) = \frac{1}{n} \sum_{i=1}^{n} [EV_i - PM_i]^2 \tag{5.5}$$

To obtain expert value, we enlisted the help of a group of recognised Hindi lexicographers who are experts in linguistics, the Hindi language, and fuzzy logic. Every expert in the room assigned a value between 0 to 1 to the relationship between each pair of words. An interval value expressing the strength between words is examined, according to the experts' own knowledge and perception. As we know, it is impossible to get values for billions of relations from experts, so we randomly selected 60 relations between nouns, 50 relations between verbs, 20 relations between nouns & verbs and 20 relations between nouns & adjectives. Finally, MSE for relations between nouns, between verbs, between noun & verbs and nouns & adjectives are shown in Table 5.2 and their graphical representation is shown in Fig. 5.10.   The results demonstrated for the selective words pair proposed method (IVFHWN) is providing good results. The proposed method can make the concept of fuzzifying Hindi WordNet feasible for all applications of natural language processing.

In literature, classical Hindi WordNet is widely used for Word Sense Disambiguation (WSD) [44, 4] so, we applied the proposed concept of IVFHWN to perform WSD at heath domain tagged dataset by IIT Bombay. To achieve WSD, centrality measures including fuzzy local and global graph connectivity measures, are calculated on the WordNet graph. These fuzzy local and global graph connectivity measures are proposed by Jain at. al. and used by many authors [5]. We applied type reduction and defuzzification on the computed intervals for the relation between two Hindi words. To evaluate the proposed system, we used recall, precision and their harmonic mean F-measure matrices. Since the proposed method offers an answer for all ambiguous inputs so in our case recall, precision and f-measure are the same.

For the comparison, we used three lexicons: Hindi WordNet, Fuzzy Hindi WordNet (where membership values are taken from linguistic experts) and Proposed IVFHWN (where membership values are automatically computed using word embeddings and interval value fuzzy logic). The comparison results on a subset of the health corpus are shown in Table 5.3. In this table, the column "All" and column "poly" shows the result on all words (polysemous and monosemous) and results only on the subset of polysemous words, respectively. Graphical representation of comparison is shown in Fig. 5.11. As can be seen in the results, Fuzzy Hindi WordNet outperforms Classical Hindi WordNet. This is because Fuzzy Hindi WordNet handles the real-life uncertainties and understands the words better while performing WSD. Next, it is also visible that fuzzy Hindi WordNet performs even better than the proposed IVFHW. The reason for this is the membership values in Hindi fuzzy WordNet are taken from various linguistic experts. In IVFHW, we automatically computed the membership values. The difference between the results is not very high, and the proposed system provides the ability to fuzzy a large number of words in a corpus which is not possible in the case of the involvement of linguistic experts.

In the proposed IVFHW, out of local and global measures, local measures are providing better results. In local measure, Degree and page rank are proving better results than the remaining three. In global edge density and compactness measures are proving better results than graph entropy.

Table 5.1. Sample sentences for the Hindi corpus with the Fuzzy relationships

	Hindi Sentences	**Pre-Processes Open class words**	**Identified Fuzzy Relationship between open class words**
1	शतरंज, सुडोकू पहेलिया आदि जैसे खेल खेले	शतरंज, सुडोकू पहेलिया. खेल	पहेलिया Fuzzy Hypernymy खेल
2	अपने विचारो को डायरी मे लिख ले	विचार, डायरी	विचार Fuzzy Meronymy डायरी
3	दिमाग़ शक्ति को बड़वा देने के लिए दवाईया जैसे ब्रह्मी, अश्वगंधा और कैलामस लें	दिमाग़, शक्ति, दवाईया, ब्रह्मी, अश्वगंधा कैलामस,	दवाईया Fuzzy Hypernymy ब्रह्मी, अश्वगंधा कैलामस
4	गहरी साँस लेने का अभ्यास करें	गहरी, साँस	गहरी, Fuzzy Attribute साँस

Table 5.2 Mean Square Error (MSE) for relations between nouns, between verbs, between noun & verbs and Between noun & adjectives

	Word	**Word**	**Total number of relations**	**#Relations considered**	**MSE**
Relation Between Nouns	No. of Noun (N) in dataset: 52230	No. of Noun (N) in dataset: 52230	N×N=2.72 Billion	60	0.63×10^{-2}
Relation Between verbs	No. of Verbs (V) in dataset: 24291	No. of Verbs (V) in dataset: 24291	V×V = 0.59 Billion	50	0.136×10^{-2}
Relation Between Noun and Verbs	No. of Noun (N) in dataset: 52230	No. of Verbs (V) in dataset: 24291	N×V=1.26 Billion	20	0.146×10^{-2}
Relation between Noun and Adjectives	No. of Noun (N) in dataset: 52230	No. of adjectives (Adj) in dataset: 22699	N×Adj= 1.18 Billion	20	0.13×10^{-2}

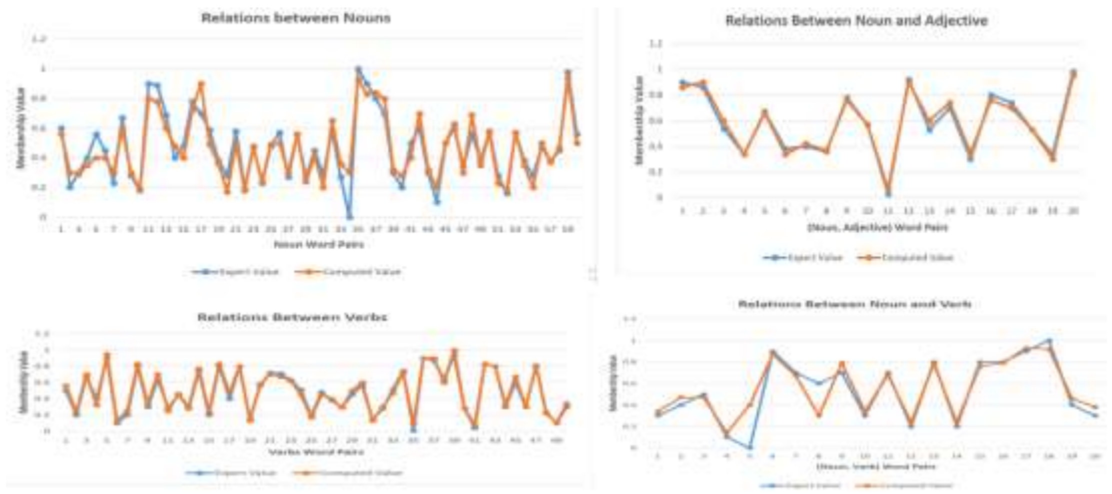


Fig. 5.10.Plots of Mean Square Error (MSE) for relations between nouns, between verbs, between noun & verbs and nouns & adjective

Table 5.3 Connectivity measures performance (F1 measure)

		Classical Hindi WordNet		**Fuzzy Hindi WordNet/ Value taken from linguistics Expert**		**Proposed IVFHW**	
		All	Poly	All	Poly	All	Poly
Local	Fuzzy Degree	0.53	0.41	0.62	0.49	0.61	0.48
	Fuzzy PageRank	0.53	0.40	0.61	0.49	0.58	0.48
	Fuzzy HITS	0.45	0.32	0.53	0.44	0.47	0.41
	Fuzzy KPP	0.47	0.37	0.56	0.45	0.49	0.44
	Fuzzy Betweenness	0.49	0.37	0.59	0.49	0.57	0.46
Global	Fuzzy Compactness	0.42	0.30	0.49	0.40	0.43	0.39
	Fuzzy Entropy	0.41	0.28	0.45	0.31	0.37	0.29
	Fuzzy Edge Density	0.43	0.30	0.51	0.42	0.46	0.35

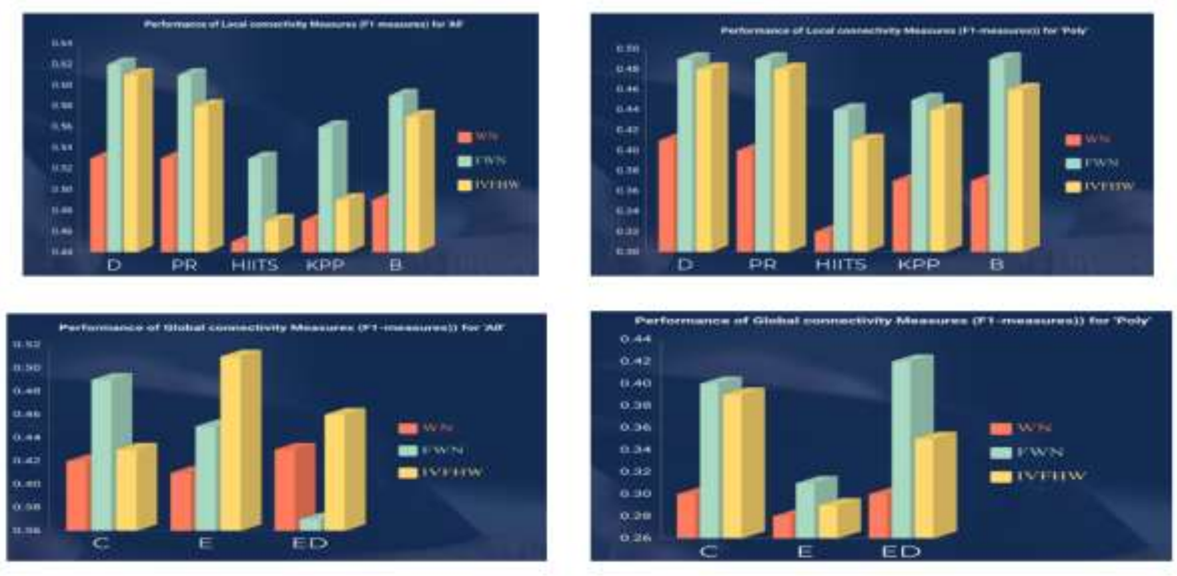


Fig. 5.11. Graphical representation of performance of local and connectivity measures for "All" and "Poly" using Hindi WordNet, Fuzzy WordNet and proposed IVFHW

5.3. Summary

Hindi WordNet is an extremely useful resource for dealing with Natural Language Processing challenges. In literature, it is confirmed that fuzzification of Hindi WordNet shows better results than classical Hindi WordNet. But in existing work, membership values for relations are taken from experts, which is not feasible in millions of words. This chapter introduces the automatic assignment of membership values to the relations between two Hindi words using lexico-syntactic patterns and various word embeddings. This automation reduces the expert's requirement and makes it feasible for any number of words. This work proposes the Interval- Valued Fuzzy relations for Hindi WordNet, which handles the uncertainty of type 1 fuzzy Hindi WordNet. For the experimentation, the proposed Interval- Valued Fuzzy Hindi WordNet is evaluated using a heath domain corpus and The results verified that (IVFHWN) is providing good results. The proposed concept is also compared with classical HWN, Fuzzy Hindi word for word sense disambiguation problem of NLP. We found IVFHWN gives better results than classical HW and FHWN. Since WordNet exists for several languages, for instance IndoWordNet (for Indian languages), EuroWordNet etc., establishing and evaluating the performance of extending WordNet to Interval-Valued WordNet with respect to these resources is an intriguing future direction. For many lexicons such as sentiwordnet, WordNet is used as a based, so extension of sentiwordnet to

Interval- Valued Fuzzy sentiwordnet can improve resource performance. It would be interesting to see how much accuracy improved by using IVFHN In the future IVFHW can be used for the other purposes of NLP, including Information keyphrase extraction, Text summarisation, query expansion etc.. To handle uncertainty in more realistic way (more close to real world)The incorporation of type-3 fuzzy logic in Hindi WordNet can be explored in future.

Chapter 6

Domain-Specific Lexicons for Sentiment Analysis

In literature, a widely used approach for sentiment analysis is the lexicon-based approach. A sentiment lexicon is an important resource, composed of sentiment words with their polarity and sentiment score (sentimental strength). Instances for sentiment lexicons are SentiwordNet [96], SenticNet [99], Sentistrength [100] etc. And for Hindi language HindiSentiWordNet [4]. In existing lexicons, the sentiment of a term not static, rather depends on the context or domain in which the term is used, i.e., its contextual semantics. Table 6.1. illustrates five instances where the polarity of words changes according to the domain. First word is "लंबा" (*laṃbā, length*) which has negative polarity in the movie domain but positive polarity in the domain of product, specifically shampoo or hair treatment products. Similarly, the word "तांबे" (*tāṃbe, copper*) has neutral polarity in the case of utensils; on the other hand, very negative polarity in the domain of jewellery. From Table 6.1, it is clearly observable that assigning static polarity to words can degrade the performance of lexicon-based sentiment analysis methods, and domain-specific sentiment lexicon can be a valuable resource in the sentiment analysis field. In this chapter, various proposed approaches for domain specific lexicon for sentiment analysis is discussed

The rest of the article organizes as follows: section 6.1 describes the proposed lexicon generation, sentiment classification method for low-resource language and experimental results.. In section 6.2, one more method based on conceptNet is described.

6.1 Proposed Methodology For Generation Of *Doslex* And Lexicon-Based Sentiment Analysis

India is a multi-lingual country and Indians feel comfortable to express their feelings in their native/local language. The number of Indian language internet users across India has reached to around 500 million in 2021[1]. It implies Indians are preferring their local languages over English to read, write and converse with each other as they feel

Table 6.1. Sample sentences having words with different polarity in different domains

	Domain	Sentence	Word with different polarity	Polarity
1.	Movie	फिल्म अनावश्यक रूप से **लंबी** है	लंबा	Negative
	Product (Shampoo)	यह शैम्पू लगाने से बाल **लंबे** हो जाते है		Positive
2.	Kitchen(Utensils)	इस बर्तन मे **तांबे** की मिलावट है	तांबे	Neutral
	Jewelry	इस हार मे **तांबे** की मिलावट है		Negative
3.	Restaurant	रेस्टोरेंट का खाना **ठंडा** था	ठंडा	Negative
	Product (Air Conditioner)	एर कंडीशनर अच्छे से **ठंडा** कर रहा है		Positive
4.	Electronics	फोन **आसानी से** चार्ज हो रहा है	आसानी से	Positive
	Movie	फिल्म मे आगे क्या होगा **आसानी से** अनुमान लगाई जा सकता था		Negative
5.	Toy	दोनो कंपनी की कारे **एक जेसी** थी	एक जेसी	Neutral
	Vehicle	दोनो कंपनी की कारे **एक जेसी** थी		Negative

more connected and heard. With the ease of internet access, Indian languages such as Hindi, Tamil, Telugu, Bengali, Kannada, and Marathi (to name just a few of the country's 22 official languages) have entered the digital space in a big way. Now it is possible for Indian users to create and consume content in their local language and there is remarkable raise in content in local languages. Many multi-nationals companies are also investing in research and resource development for local languages. Indian government also promoting study and work on regional languages. Despite of a large amount of content and encouragement there is lack of existing resources in Indian languages. Developing resources for local/regional languages will usher in an era where there will be an economic boost to the Indian economy. In this work, we propose a method that creates a resource that is domain-specific lexicon for sentiment analysis for low-resource Indian languages.

This lexicon generation framework considers six main steps shown in Fig.6.1. For the better understanding and ease of explanation of the proposed methodology, the illustrations are given for Hindi language.

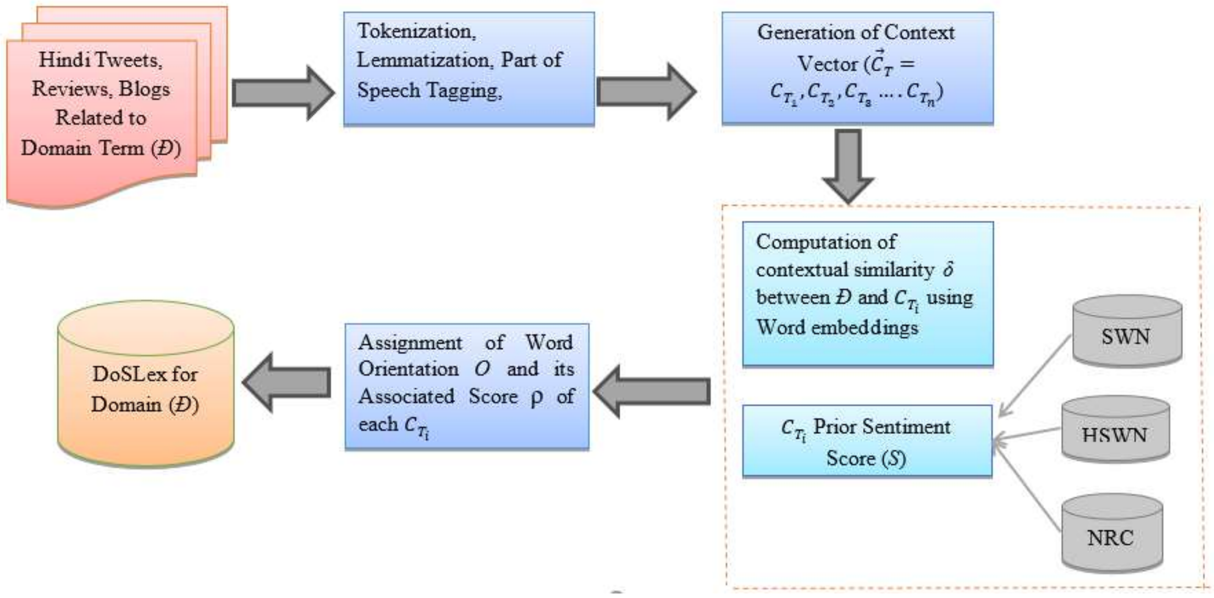

Hindi Tweets, Reviews, Blogs Related to Domain Term (Đ)
Tokenization, Lemmatization, Part of Speech Tagging,
Generation of Context Vector ($\vec{C}_T = C_{T_1}, C_{T_2}, C_{T_3} \dots C_{T_n}$)
Computation of contextual similarity δ between Đ and C_{T_i} using Word embeddings
C_{T_i} Prior Sentiment Score (S)
SWN
HSWN
NRC
Assignment of Word Orientation O and its Associated Score ρ of each C_{T_i}
DoSLex for Domain (Đ)

Fig. 6.1. Overall Framework for Hindi *DoSLex* Generation

6.1.1 Data Collection and Pre-Processing

The proposed approach to build lexicon can be applied to any domain but in this chapter to make it feasible, we have selected five Hindi domains, one Tamil and one Bangla namely movie, restaurant, politics, books and tourism, Tamil movie and Bangla news. Sentences related to these domains are extracted from various sources including Twitter, News websites, IT Bombay site. After extraction, first pre-processing step is to lemmatize and tag extracted data according to suitable

ALGORITHM 6.1 Domain Specific Lexicon (*DoSLex*) Generation

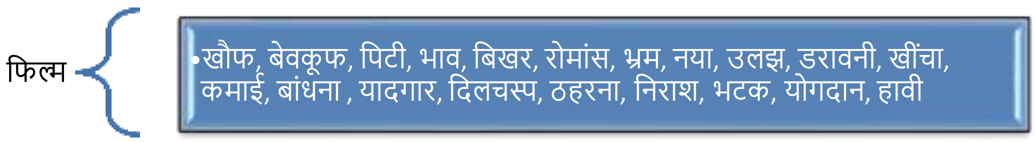
```
Input: Dataset related to domain (Đ) from various sources for language L
Output: Domain-Specific Lexicon DoSLex comprising polarity and sentiment score for each
        word. (classification of words in four polarity: positive, Very positive, Negative and
        Very   Negative)
    Initialize $POS^{+}$ List, $VPOS^{++}$ List, $NEG^{-}$ List, $VNEG^{--}$ List  to empty
    for each sentence in dataset
        Perform Lemmatization and Tokenization
        for each token do
            Part-of-Speech tagging (POS)
            if POS=  nouns||adjectives|| adverbs || verbs || Unknown then
                Add token in to context vector $(\overrightarrow{C_T} = C_{T_1}, C_{T_2}, C_{T_3} \ldots . C_{T_n})$
    for each context term $C_{T_i} \in \vec{C}_T$
        Calculate  $\delta$ (Đ, $C_{T_i}$)   //  $\delta$ is semantic relation between domain term and context term
        Calculate  $S(C_{T_i})$  //  $S$ is prior sentiment of context term
        Plot in 2D circle
        $A_i = \delta$ (Đ, $C_{T_i}$)      // radius  $0 \leq A_i \leq 1$
        $\emptyset_i = S(C_{T_i}) * \pi$   //angle
    Coordinates for each context term
        $X_i = A_i \sin \emptyset_i$  // polarity score of context term
        $Y_i = A_i \cos \emptyset_i$       // polarity of context term
    if $(X_i, Y_i)$ lies in first quadrant
        Add $C_{T_i}$ in  $POS^{+}$ List
    Elseif $(X_i, Y_i)$ lies in Second quadrant
        Add $C_{T_i}$ in  $VPOS^{++}$ List
    Elseif $(X_i, Y_i)$ lies in Third quadrant
        Add $C_{T_i}$ in  $VNeg^{--}$List
    Elseif  $(X_i, Y_i)$ lies in Fourth quadrant
        Add $C_{T_i}$ in  $Neg^{-}$ List
    Return DoSLex,  $POS^{+}$ List, $VPOS^{++}$ List, $NEG^{-}$ List, $VNEG^{--}$ List
```

part of speech. For lemmatization, we have used pyspark from John snow labs and for part of speech tagging NLTK POS tagger is used.

6.1.2 Generation of context vector $(\vec{C}_T)$

for part of speech tagging NLTK POS tagger is used.

Context vector is denoted as ($\vec{C}_T = C_{T_1}, C_{T_2}, C_{T_3} \ldots . C_{T_n}$) where, $C_{T_i} \in \vec{C}_T$ and C_{T_i} is the context term. Context terms are words, related to domain *Đ* whose polarity we are going to assign using proposed approach. This work considers all terms that occur in the same context to generate lexicon for each domain *Đ*. Let's say if domain *Đ* is "फिल्म" (Movie), then all the sentences related to domain are extracted from different platforms (in previous step). This implies that context vector comprises contextual terms of domain. For domain "फिल्म" a sample context vector is shown in following figure


फिल्म
•खौफ, बेवकूफ, पिटी, भाव, बिखर, रोमांस, भ्रम, नया, उलझ, डरावनी, खींचा, कमाई, बांधना , यादगार, दिलचस्प, ठहरना, निराश, भटक, योगदान, हावी

6.1.3 Computation of Semantic relation score (δ) between *Đ* and C_{T_i} using word embeddings

To compute semantic relation (**δ**) between domain *Đ* and CTi , we have utilized the word embedding method. A word embedding is a term used for vector representation of words and phrases with similar words and phrases having similar word representation in predefined vector space [101]. FastText is one of the effective word embedding for Hindi language. FastText is an open-source, free, lightweight library that allows users to learn text representations and text classifiers. FastText (essentially an extension of the word2vec model) treats each word as if it is composed of character n-grams. That means the vector for a word is made up of the sum of its character n-grams. For this proposed work, we initially applied FastText embeddings, but during experimentations, the results were not satisfying, so we shifted to M-Bert (Multilingual BERT) and then finally to the latest embedding method for Indian languages MuRIL (Multilingual Representations for Indian Languages) Multilingual Bidirectional Encoder Representations from Transformers (M-Bert) is a pre-trained language model developed using the 104-language Wikipedia corpus [102]. The main accomplishment of this model is its ability to transfer across languages with different vocabulary, such English and Devanagari Hindi, despite the fact that it was trained on distinct 104 monolingual corpora beforehand. Additionally, empirical evidence suggests that M-Bert does a good job at detecting semantic relationships between words. Another model built on Bert that is pre-trained in Indian languages is MuRIL [103]. It is a recently released BERT model that has been pre-trained on 17 Indian languages, including Hindi and English, as well as their transliterated equivalents. It is a modified version of M-Bert for Indian languages with limited resources because it trains using pairs of translated and transliterated texts.

6.1.4 Prior generic sentiment score (S)

In this step, each context word $C_{T_i} \in \vec{C}_T$ assigned with an initial score from external sentiment lexicons. Here sentiment for each Hindi context word is taken from the Hindi SentiWordNet lexicon, NRC-Emotion lexicon [104, 105] and English SentiWordNet. For maximum coverage of words, more than one lexicon is considered. Those words not available in Hindi Senti WordNet and NRC emotion lexicon are translated to English and checked in English Senti WordNet. For Bangla and Tamil languages, Bengali SentiWordNet and Tamil SentiWordNet is utilized.

6.1.5 Computation of word orientation *O* and its associated sentiment score ρ of each C_{T_i}

Each domain *Đ* and its context term $C_{T_i} \in \vec{C}_T$ are represented in a geometric 2D circle. Circular representation is beneficial as it allows us to use trigonometric identities to estimate the sentiment orientation and strength of the various context terms. The proposed *DoSLex* is based on contextual semantic relation of domain *Đ*. In this geometric circle, domain *Đ* is centre of the circle and other context terms of domain *Đ* are pointed around (shown in Fig.6.2). Position of context term C_{T_i} is defined jointly using word embeddings and prior sentiment score. Mathematically, 2D circle polar coordinate system is represented with the following equation:

$$A^2 - 2AA_0 \cos(\emptyset - \theta) + A_0^2 = R^2 \qquad (6.1)$$

Where, R is denoted as radius of the circle, polar coordinate of domain term *Đ* that situated in centre of the circle is (A_0, θ) and polar coordinate of context term C_{T_i} is denoted as (A, $\emptyset$). This study considers context word C_{T_i}, radius A_i and angle $\emptyset_i$ to build *DoSLex* of *Đ* and Prior of sentiment score S and word embedding semantic relationship score δ [10] of word C_{T_i} . Below equations help to build 2D circle for *DoSLex*.

$$A_i = \delta\,(Đ, C_{T_i}) \qquad (6.2)$$

$$\emptyset_i = S(C_{T_i}) * \pi \qquad (6.3)$$

Radii of all terms in 2D circle consider scale between 0 to 1 and radius R of circle is 1. All angle values are computed in radian.

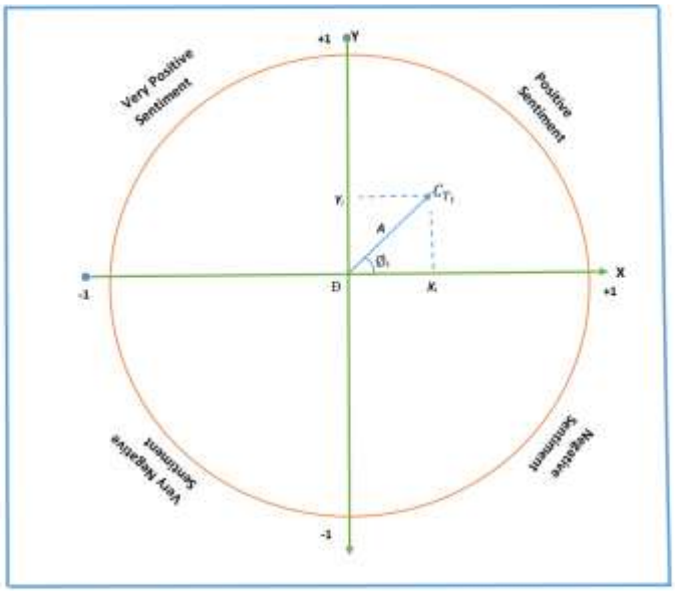


Fig. 6.2. 2D circle representation for *DoSLex* with Four Polarity quadrants

6.1.6 Generation of *DoSLex*

DoSLex uses the Cartesian coordinate system from the polar coordinate system by applying sine and cosine trigonometric functions to compute coordinates of term C_{T_i} .

$$X_i = A_i \sin \emptyset_i$$

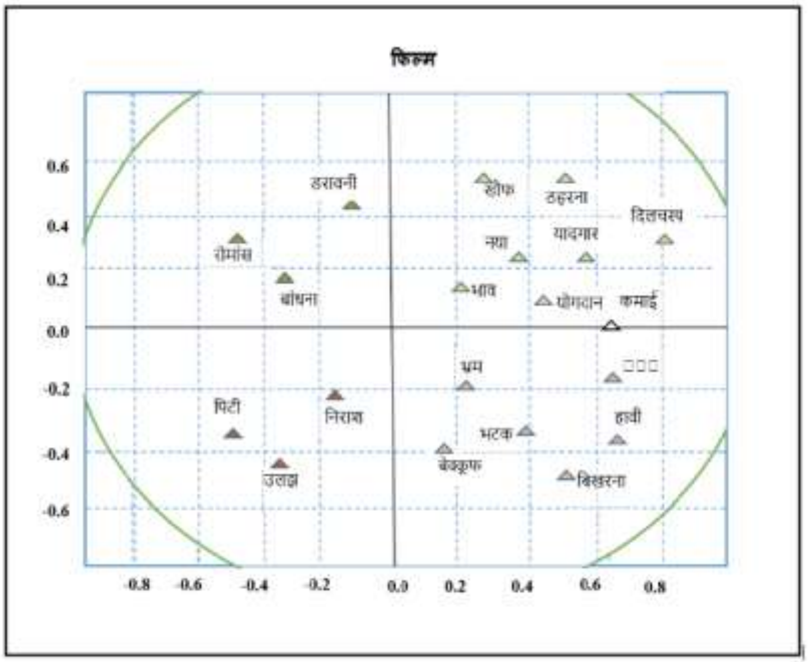
$$Y_i = A_i \cos \emptyset_i$$

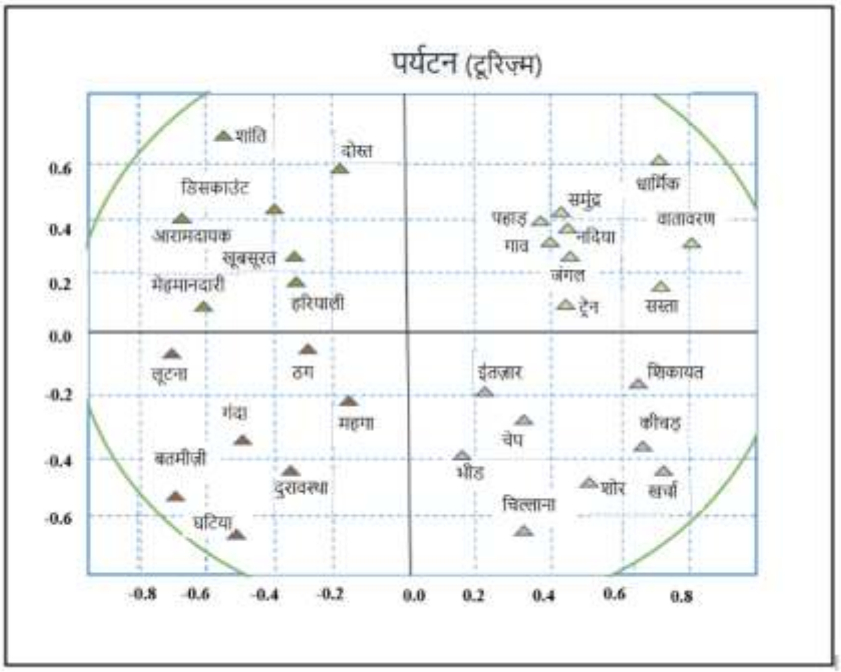
Y-axis of the Cartesian coordinate system denotes sentiment orientation (positive, very positive, negative and very negative) of term C_{T_i} and X-axis denotes the degree (score) of the C_{T_i} term in the context of the domain. Positive Y value defines positive sentiments and negative Y value defines negative sentiments. A small X value denotes strong sentiment.

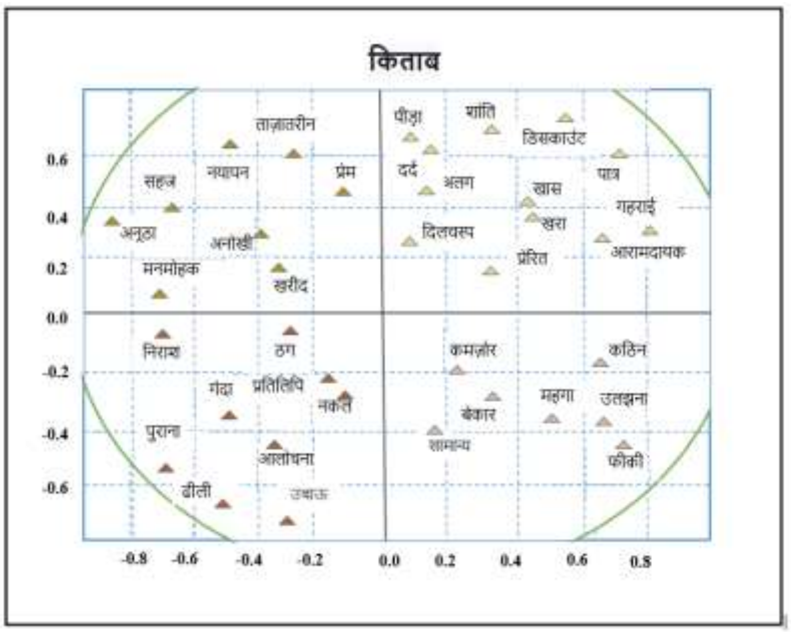
Fig. 6.3 represent a sample 2D circle representation *DoSLex* of domain movie, tourism, books and news for the Hindi language. In this figure, terms present in the first and second quadrant are positive terms with respect to domain. And terms in the third and fourth quadrant are negative terms with respect to domain. This Domain-Specific lexicon is a very significant resource for research in sentiment analysis. It provides context words strongly related to the domain with polarity and ignores context words that contribute low in polarity towards the domain.

6.1.7 Lexicon-based Sentiment Analysis using proposed *DoSLex*

In this section, we discussed a methodology for lexicon-based sentiment analysis using proposed *DoSLex* for Hindi language. Fig. 6.4. presents the overall workflow diagram for sentiment analysis task. As for sentiment score computation, first step is to pre-process the extracted text T. Input text can be taken from any source such as reviews, Tweets etc. Pre-processing includes removal of URLs, abbreviations, punctuations and symbols. Next step is identification of domain using topic modeling LDA (Latent Dirichlet Algorithm). The identified domain is then searched in to proposed list of lexicons (for instance in this proposed we have created lexicons for movies, tourism, news, politics, books domain). In case of non-availability of domain, system checks it's synonymy in *DoSLex*. And if synonymy is also not available then algorithm 1 is applied to generate required lexicon. After getting the relevant domain specific lexicon


फिल्म
डरावनी
ख़ौफ
ठहरना
दिलचस्प
रोमांस
नया
यादगार
बांधना
भाव
योगदान
कमाई
भ्रम
□□□
पिटी
निराश
हावी
भटक
बेवकूफ
उलझ
बिखरना
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
0.8


पर्यटन (टूरिज़्म)
शांति
दोस्त
डिसकाउंट
धार्मिक
समुंद्र
पहाड़
वातावरण
आरामदायक
नदिया
गाव
खूबसूरत
जंगल
मेहमानदारी
हरियाली
ट्रेन
सस्ता
लूटना
ठग
इंतज़ार
शिकायत
गंदा
महंगा
कीचड़
चेप
बतमीज़ी
भीड
दुरावस्था
शोर
खर्चा
चिल्लाना
घटिया
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
0.8



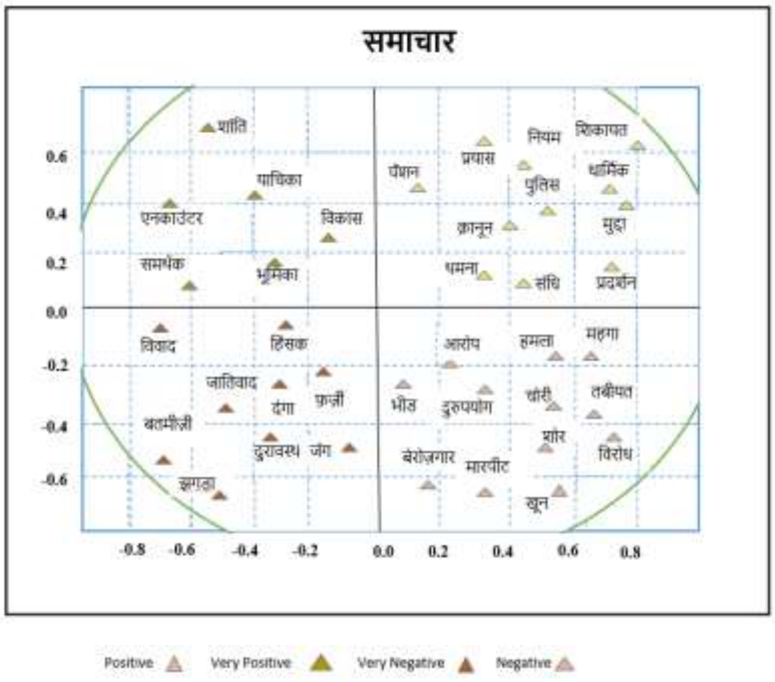
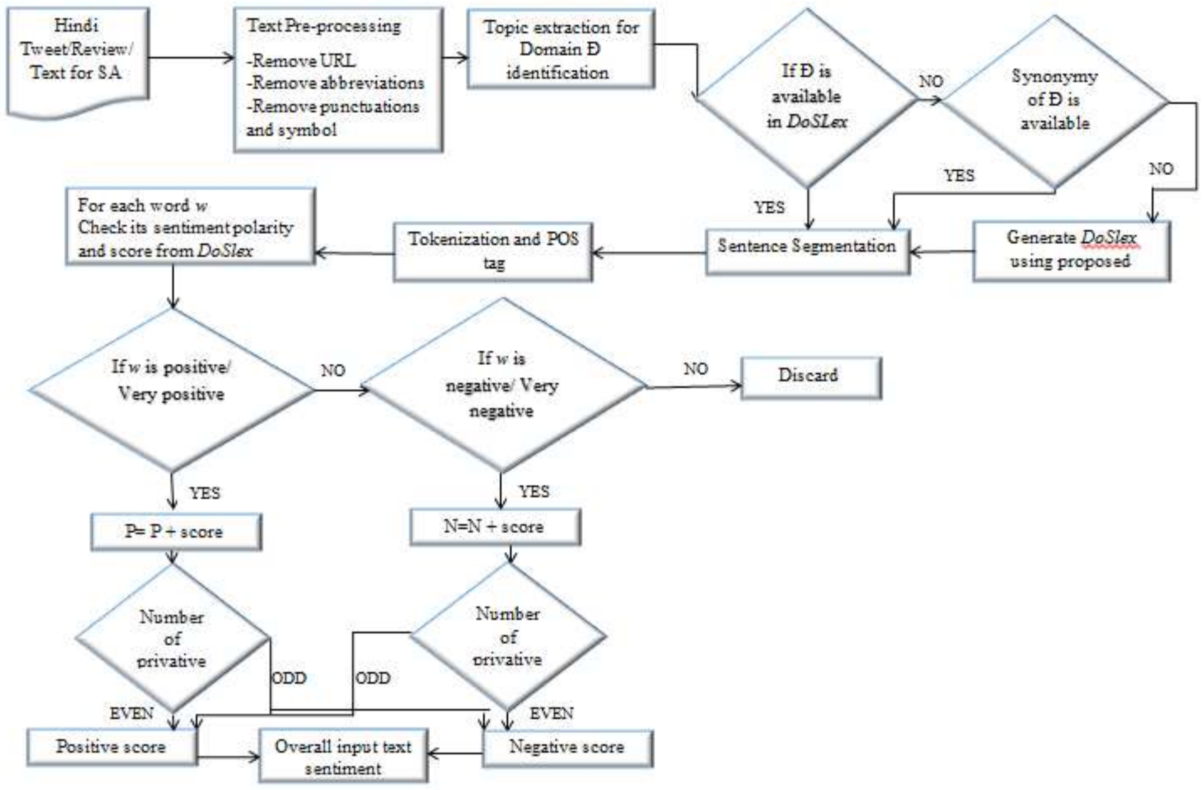


Fig. 6.3. *DoSLex* 2D circle representation for domains फिल्म (movie), पर्यटन (Tourism), किताब (book) and समाचार (news) in Hindi

next step is spilt the text in to sentences ($T = S_1, S_2, S_3, \ldots S_p$) where T is the text and $S_i \epsilon T$ is the sentence of text. Each sentence is then tokenized and each token tagged according to appropriate part of speech. For each noun, adverb, adjective and verbs tag we check the sentiment polarity and strength in *DoSLex*. So now, there are two categories of sentiment score: one is positive and one is negative. The initial score of each category is zero. If there is odd number of privative, the sentiment score falls in to another category. And in case of even number of privative the previous sentient score remains same. In last step all the sentiment scores are aggregated thus we have overall positive score/negative score of sentence.



Hindi Tweet/Review/ Text for SA
Text Pre-processing
-Remove URL
-Remove abbreviations
-Remove punctuations and symbol
Topic extraction for Domain Đ identification
If Đ is available in DoSLex
NO
Synonymy of Đ is available
YES
NO
YES
Generate DoSlex using proposed
Sentence Segmentation
Tokenization and POS tag
For each word w Check its sentiment polarity and score from DoSlex
If w is positive/ Very positive
NO
If w is negative/ Very negative
NO
Discard
YES
YES
P= P + score
N=N + score
Number of privative
Number of privative
ODD
ODD
EVEN
EVEN
Positive score
Overall input text sentiment
Negative score

Fig.6.4. Overall flowchart for lexicon-based sentiment analyses approach using *DoSLex*

6.1.8 EXPERIMENTAL RESULTS AND DISCUSSIONS

This section performs several experiments to test the efficiency of the proposed method. This section includes evaluation matrices, datasets description, experiment details with results, discussion.

6.1.8.1 Dataset Description and Evaluation Matrices

Getting annotated dataset for low-resource languages is a challenging task. Although the proposed approach can be applied to any language but for ease and understandability, three low resource Indian languages Hindi, Tamil and Bangla are used. This section discusses the dataset for Hindi, Bangla and Tamil languages. Hindi Dataset by IIT Bombay: Annotated dataset for domain tourism and movie in Hindi language is taken from IIT B site. Movie domain consist of 484 positive and negative labelled reviews [13]. Tourism domain has 200 positive and negative reviews [106].
Twitter Dataset: The dataset for evolution of proposed methodology has been fetched from twitter using tweepy API. The 2467 tweets have been extracted for “राजनीफत” (politics) domain from twitter by selecting language Hindi in the search filter. Out of these tweets, we removed tweets with English words written in Hindi, non-Hindi tweets and tweet without subjectivity. After elimination 1534 subjective tweets were left. These tweets are given to three annotators for manual labelling. These annotators are the graduate students of university and native speakers of Hindi language. They have been asked to label each tweet positive or negative according to their understanding. Now, different annotators may have different perceptions of meaning convey through context. In case of conflict on the sentiment of the same word kappa inter-annotator agreement is used for resolution. It considers the possibility of agreement. At last dataset of 1534 tweets of domain “politics” are extracted in which 654 are positive tweets and 880 are negative tweets.
Amazon Product review dataset: It is a benchmark dataset which is widely used for sentiment analysis task. All the reviews are collected from amazon.com. There are 1000 positive 1000 negative reviews for each product. We collected for domain “किताब” (books) and translate them using Google translator.
News Comment Dataset for Bangla: This dataset is collected from two popular Bangla newspapers, BBC Bangla and Prothom Alo. It contains 5205 positive comments and 5600 negative tagged comments [107] Tamil Movie Reviews: This dataset consists of 200 positive and 200 negative comments reviews from Movie dataset (Film:

Baahubali) [108]. Table 6.2 shows a detailed description of annotated dataset of all domains.

The evaluation matrices consider in this research are precision (P_R), recall (R_E) and F-measure score (F_M). These matrices will assess the performance of the proposed Domain-Specific lexicon. The P_R is the fraction of correctly classified sentence of a sentiment (Positive or negative) among number of sentences classified of the relevant sentiment. The R_E is fraction of correctly classified sentence count among the total number of sentences of that sentiment. The R_E and P_R helps to understand the degree of relevance. The F score is the harmonic mean of the P_R and R_E.

6.1.8.2. Results Analysis of Proposed *DoSLex* on Various Datasets

This section explains the detail experimental results that will help to comprehend how the proposed Domain-Specific lexicon performed well in sentiment classification task. For evaluation, the lexicon based approach has been applied on different domains. We compute the performance of proposed *DoSLex* with different source of prior sentiment knowledge bases. First is *DoSLex* with HSWN only. Second is *DoSLex* with HSWN+NRC and last one is *DoSLex* with HSWN+NRC+ESWN. Table 6.3 presents the evaluation of all the cases. It can be noticed from the table that *DoSLex* with HSWN achieves maximum F-measure of only 0.306 in domain of Hindi movies. The performance of *DoSLex* with HSWN is very low due to narrow coverage of polarity words in Hindi SentiWordNet related to the domain HSWN has very limited polarity words in the lexicon. Out of these polarity some words are wrongly labelled for example word "हिंसा" hinsa is marked with neutral polarity instead of negative polarity. However, *DoSLex* with HSWN+NRC gives maximum F-measure of 0.542 due to increase in polarity word coverage and reconsideration of neutral words. Out of these three cases, *DoSLex* with HSWN+NRC+ESWN (FastText) achieves a better F-measure of maximum 0.683 in domain movie. This performance depends on prior sentiment values and can be improved further with a more accurate and large size of sentiment lexicon. The proposed approach is also reliant on the performance of word embeddings including FastText. M-Bert and MuRIL. Table 6.3, 6.4, & 6.5 provides the results of all combinations with three word embeddings. Fig.6.5 represents the f-measure comparison of FastText, M-Bert and MuRIL embeddings on different domain datasets. Figures shows M-Bert performs better than FastText because in FastText each word has fixed representation regardless of context and M-Bert is a contextual based word embedding. MuRIL performs best out of these three embedding techniques. MuRIL has similar training strategy as M-Bert with few modifications Maximum F-measure is 0.837 given by *DoSLex* +HSWN+NRC+SWN with MuRIL embedding. Next experiments are done for Bangla and Tamil languages for news and movie domain. Table 6.6shows results with different combinations. For Bangla news domain *DoSLex* + BSWN + MuRIL combination provided best performance with 0.68 F-measure. And for Tamil movie domain *DoSLex*+ TSWN+ MuRIL gives best results of 0.711 f-measure. BSWN and TSWN are Bangla SentiWordNet and Tamil SentiWordNet respectively. The results are shown in graphs in Fig. 6.6. The standard/classic Indic_sentiwordNet (SentiWordNets for Indian languages such as Hindi, Bangla, Tamil) are compared with proposed *DoSLex*. The comparison with existing popular generic-lexicons is performed on different domains to show the performance of proposed *DoSLex* over existing lexicons. For experimentation, sentiment classification method is applied to all datasets. Table 6.7 and Fig. 6.7. shows the evaluation results and from the table it can be noticed that proposed *DoSLex* performs much better with maximum F-measure of 0.837, 0.680, 0.711 in Hindi, Bangla and Tamil languages respectively. The major reason of very low performance of generic lexicons is non consideration of topic or domain of the text. For instance, “small” is assigned negative in SentiWordNet but it is positive in many domains such as mobile “small and handy”. Moreover, the proposed lexicon is extendable to any number of polarity words depended on prior knowledge and word similarity. And *DoSLex* can be created for any language either resource low or rich. *DoSLex* is also fully scalable and adaptable to any domain. Fig. 6.5, 6,6 and 6.7 shows the graphical representation of whole experimental results.

Table 6.2: Annotated Dataset Description

Language	Domain	Positive	Negative
Hindi	Books	1000	1000
	Movies	127 (258 sentences)	125 (224 sentences)
	Tourism	100 (256 sentences)	100 (249 sentences)
	politics	654	880
Bangla	News	5205	5600
Tamil	Movie	200	200

Table 6.3: Performance evaluation results of proposed *DoSLex* with different prior sentiment bases and FastText

		DoSLex with HSWN (FastText)			*DoSLex* with HSWN+ NRC (FastText)			*DoSLex* with HSWN+ NRC+SWN (FastText)		
Dataset	Domain	P_R	R_E	F_M	P_R	R_E	F_M	P_R	R_E	F_M
IIT B	Movie	0.314	0.299	0.306	0.563	0.523	0.542	0.689	0.679	0.683
	Tourisms	0.271	0.253	0.261	0.512	0.553	0.531	0.623	0.653	0.637
Amazon	Books	0.262	0.234	0.247	0.534	0. 434	0.478	0.644	0.634	0.638
Twitter	Politics	0.212	0.22	0.215	0.502	0.450	0.474	0.612	0.6	0.605

Table 6.4: Performance evaluation results of proposed *DoSLex* with different prior sentiment bases and M-Bert

		DoSLex with HSWN (M-Bert)			*DoSLex* with HSWN+ NRC (M-Bert)			*DoSLex* with HSWN+ NRC+SWN (M-Bert)		
Dataset	Domain	P_R	R_E	F_M	P_R	R_E	F_M	P_R	R_E	F_M
IIT B	Movie	0.332	0.326	0.328	0.643	0.646	0.644	0.734	0.767	0.750
	Tourisms	0.317	0.335	0.325	0.613	0.672	0.641	0.712	0.734	0.722
Amazon	Books	0.319	0.345	0.331	0.667	0. 656	0.661	0.745	0.756	0.750
Twitter	Politics	0.321	0.322	0.321	0.623	0.634	0.628	0.745	0.712	0.728

Table 6.5: Performance evaluation results of proposed *DoSLex* with different prior sentiment bases and MuRIL

		DoSLex with HSWN (MuRIL)			*DoSLex* with HSWN+ NRC (MuRIL)			*DoSLex* with HSWN+ NRC+SWN (MuRIL)		
Dataset	Domain	P_R	R_E	F_M	P_R	R_E	F_M	P_R	R_E	F_M
IIT B	Movie	0.399	0.419	0.408	0.773	0.753	0.762	0.829	0.836	0.832
	Tourisms	0.402	0.423	0.412	0.762	0.775	0.768	0.833	0.843	0.837
Amazon	Books	0.372	0.364	0.367	0.754	0. 774	0.763	0.824	0.835	0.829
Twitter	Politics	0.362	0.352	0.356	0.72	0.761	0.739	0.842	0.831	0.836

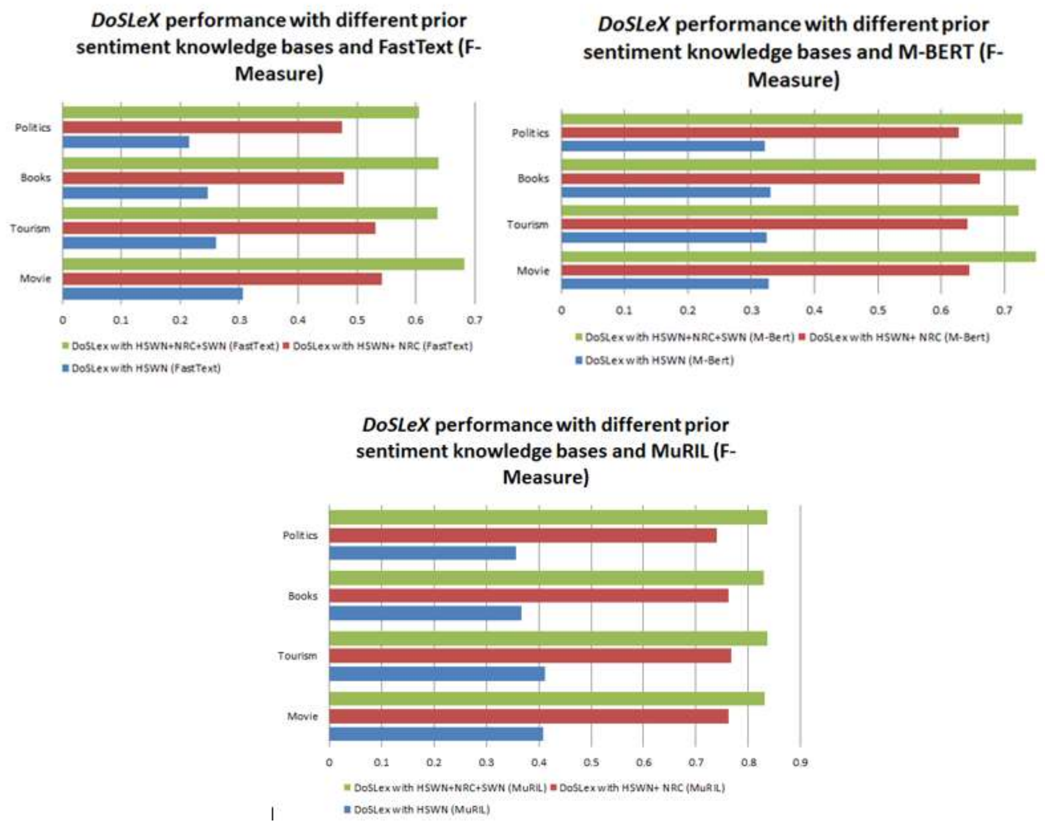


Fig. 6.5. Graphical representation for the performance of *DoSLex* with different prior sentiment knowledge bases and embeddings (F-measure)

Table 6.6: Performance evaluation of *DoSLex* for Bangla and Tamil low-resource languages

Domain & Language	Model Combinations	P_R	R_E	F_M
Bangla	*DoSLex*+ BSWN+ FastText	0.323	0.337	0.329
News	*DoSLex*+ BSWN +M-Bert	0.572	0.563	0.567
	DoSLex+ BSWN+ MuRIL	0.685	0.677	0.680
Tamil	*DoSLex*+ TSWN +FastText	0.381	0.372	0.376
Movie	*DoSLex*+ TSWN +M-Bert	0.591	0.601	0.595
	DoSLex+ TSWN + MuRIL	0.721	0.703	0.711

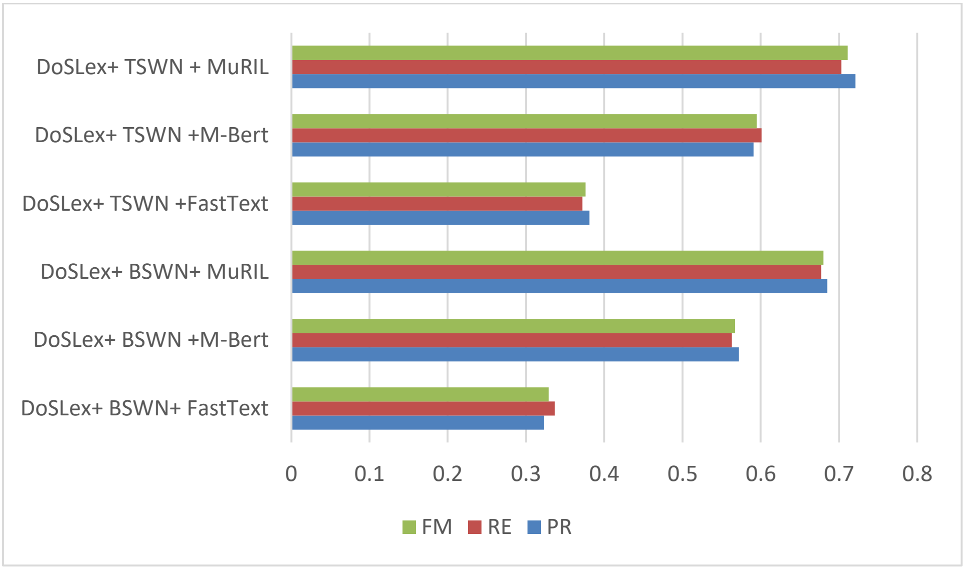


Fig. 6.6: Graphical representation for performance of DoSLex for Bangla and Tamil language

Table 6.7: Performance evaluation results of proposed *DoSLex* with Indic_SWN on various datasets

		DoSLex (Proposed)			Indic_SWN (HSWN, BSWN, TSWN)		
Dataset	Domain	P_R	R_E	F_M	P_R	R_E	F_M
IIT B	Movie	0.829	0.836	0.832	0.160	0.181	0.169
	Tourisms	0.833	0.843	0.837	0.158	0.140	0.149
Amazon	Books	0.824	0.835	0.829	0.176	0.174	0.175
Twitter	Politics	0.842	0.831	0.836	0.147	0.125	0.135
Reference	Bangla News	0.685	0.677	0.680	0.120	0.131	0.125
reference	Tamil Movie	0.721	0.703	0.711	0.061	0.134	0.083

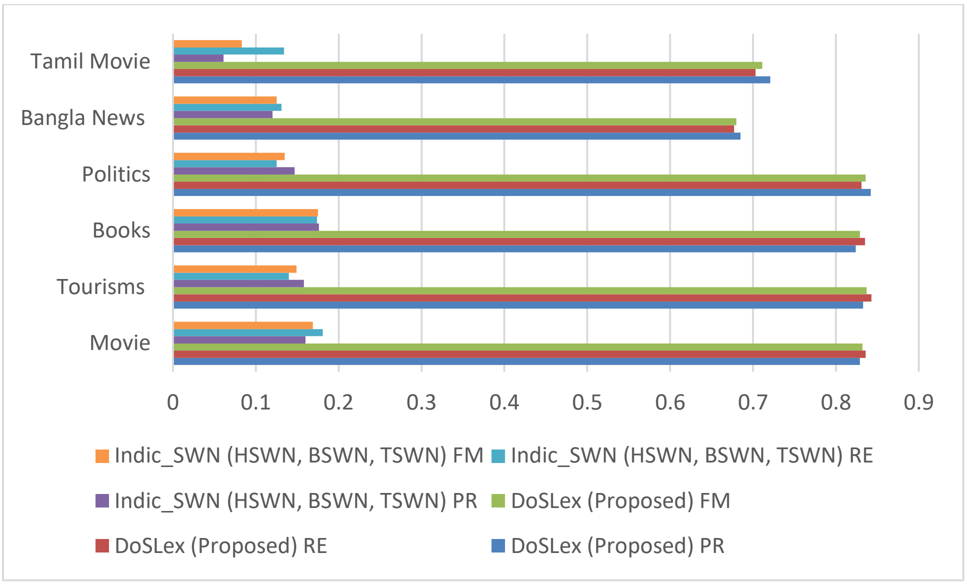


Fig 6.7. Graphical representation for comparative results of proposed *DoSLex* with indic_sentiwordnets on various datasets

6.2 Proposed Methodology: Building Domain-Specific Sentiment Lexicon using Random walk-based model on Common-sense Semantic Network

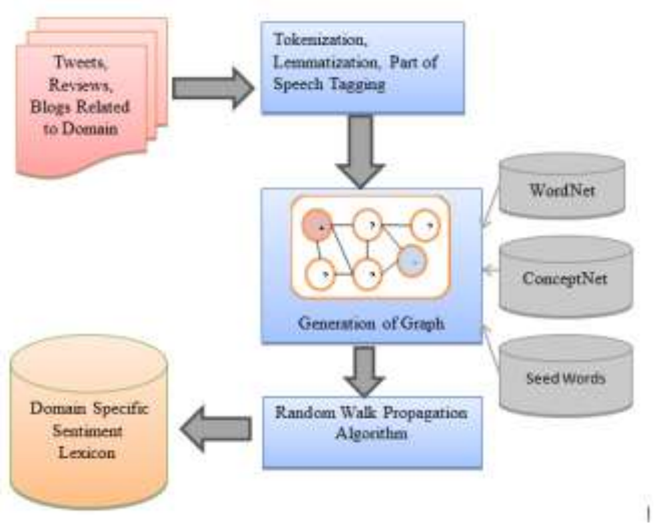

Tweets, Reviews, Blogs Related to Domain
Tokenization, Lemmatization, Part of Speech Tagging
WordNet
ConceptNet
Generation of Graph
Seed Words
Domain Specific Sentiment Lexicon
Random Walk Propagation Algorithm

Fig. 6.8 Overall Framework of Proposed Approach

The proposed approach is based on semi-supervised learning as it uses a few seed words with known polarities to enlarge the sentiment lexicon to thousands of words. The popular random walk algorithm is used and modified for this purpose. The proposed approach broadly focuses on (as shown in Fig. 6.8): 1. Collecting standard domain-specific seed words with calculated polarities. 2. Extracting ConceptNet relations for network construction and initializing their weights. 3. Collecting standard domain-specific seed words with calculated polarities. 4. Extracting ConceptNet relations for network construction and initializing their weights. 5. Training the weights using a standard gradient descent and a random walk algorithm. 6. Finally apply a random walk algorithm on the graph with trained weights for classification. 7. Plotting a graph between average polarity error and weights of ConceptNet relations. The proposed approach first constructs a network where two nodes are linked if a

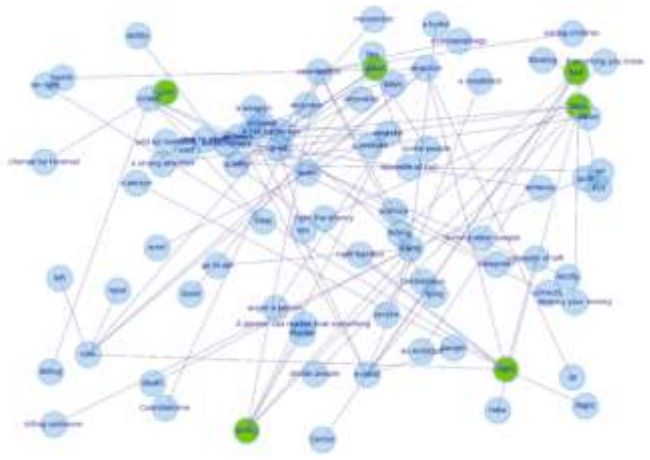


Fig. 6.9: Snippet of ConceptNet graph for seed word of domain “Hate speech”

semantic relation exists between them. This work uses ConceptNet for semantic relations it also accounts for common-sense, foreign language words and phrases i.e. multiple words for single nodes. The resulting graph G is a graph (C,E), where C is a set of concepts in ConceptNet. E is the set of edges connecting concepts/nodes in the graph. Graph G is generated using a domain-specific list of seed words (initial nodes) having known polarity. To get prior sentiment knowledge of seed words SenticNet is used. Next graph is extended to level L by adding related concepts from ConceptNet. The ConceptNet graph contains labelled (green nodes with known polarity) as well as unlabeled (Blue nodes with unknown polarity) is shown in Fig 6.9. Now, the weights of relations in graph G are trained using standard gradient descent and random walk algorithm. A Sentiment weights is provided to all relations of conceptNet. Assigning weights to relations is based on the assumption that concepts pass their sentiment value to their neighbor concepts in different ways depending on the relations connecting them. All the relations cannot have same sentiment weightage. So, we use machine learning approach, gradient descent to provide weightage to relations. Step 1: Initialize the relations weights with random values and calculate cost. We initialized the weight of 0.1 to all the direct relations and -0.1 to all inverse relation.

Inverse or negative relations are tend to change the polarity of connected nodes. In our work, cost is Average Polarity Error (APE) i.e. difference between the classified sentiment value and the reference sentiment value for all the classified words. Reference sentiment value is SenticNet value of node. Step 2: Calculate the gradient i.e. change in APE when the weights of relations are changed by a very small value from their original randomly initialized value. This helps us move the values of relations in the direction in which APE is minimized. Weights of direct relations are increased by a small value i.e. 0.01 and weights of inverse relations are also deceased by 0.01. Step 3: Adjust the relation weights with the gradients to reach the optimal values where APE is minimized Step 4: Use the new trained relation weights for sentiment lexicon generation and to calculate the new APE. Step 5: Repeat steps 2 and 3 till further adjustments to weights doesn't significantly reduce the APE.

6.2.1 Random Walk model over Weighted ConceptNet Network

After the construction of concept graph with weighted relations next is generating a sentiment lexicon by propagating know values using random walk model. An improvised random walk algorithm is applied to propagate the sentiments from labelled nodes to unlabeled nodes. A random walk starts from a blue node and may end up on a green node thus affecting the polarity of the blue node by propagating sentiments from the green node. The walk begins from an unlabeled word which may be related to other nodes in the graph with weighted edges of given magnitudes. For this we defined transition probability $P_{t+1|t}\,(j|i)$from node 1 to node *j* by normalizing the weights of the edges of node *i*

$$P_{t+1|t}\,(j|i) = \frac{c_{ij}}{\sum_{k=1}^{n}(c_{ik})}$$

Where k represents all nodes in the neighbourhood of i, $P_{t+1|t}\,(j|i)$ denotes the transition probability from node I at step t to node j at time step t+1. The length of the random walk is restricted to a length β and count of random walk are restricted to α. The walk ends if a unlabelled node is reached to labelled node. At the end of the process unlabeled word is labelled with a polarity magnitude directly proportional to the polarity magnitude of the labelled word reached at the end of the walk and to the average weight encountered in its journey. Each relation with it has an associated weight, that is if there is a directed edge of a relation between two words i and j with weight w then, if polarity of i is Pi then polarity of j i.e. pj is,

$P_j = P_i*(1+w)$

If the words i and j are connected by multiple (say n) edges then the Geometric Mean weight, w_{GM} can be calculated by taking the nth root of the Absolute value of the product of the weights of intermediate edges. For eg. if there are n intermediates edges, W_i for i = 1 to n from node(word) i to node(word) j then,

$W_{GM} = | \prod W_i , for\ i = 1\ to\ n |^{1/n}$

Further W_{Gm} is multiplied by -1 if there are an odd number of Inverse relations in the n edges.

The initialized weight could have been anything though it has to be chosen wisely so that the convergence does not get stuck in the local minima. The weights inverse relations were avoided to be taken -0.1 due to sum of probabilities restrictions.

6.2.2 Experimental Results

Astonishing results were obtained just from using the seed ['good', 0.664], ['bad', -0.36], ['Love', 0.655], ['hate', -0.83], ['right', 0.67], ['wrong', -0.76] related to hate speech domain taken from a previously generated sentiment dictionary i.e. SenticNet and related words going as deep as two layers. These six words gave more words with their predicted polarity (around 930) of which around 470 words was present in the SenticNet (which could be used for getting the Average Polarity Error needed for weights training). The dictionary size increased from 6 to 930 using a small amount of training data where the polarities are accurate for 78.801 % words. The process was repeated for many iterations for training the weights with the objective of reducing the Average Polarity Error. Initially the Average Polarity Error was around 0.482 and it started to decrease as the iterations succeeded. In the results the classification accuracy may not go hand in hand with Average Polarity Error in the short run because of the way the accuracy is defined. Accuracy is defined as percentage of positive and negative words correctly predicted with polarity greater and less than zero respectively. This is the reason Average Polarity Error was chosen as the objective function instead of classification accuracy. But in the long run an inverse relation between Average Polarity Error and classification accuracy was observed. There has been related work done in which some of the ConceptNet relations were used but what we here do is use all the relations and whatever relation should be chosen are automatically extracted when

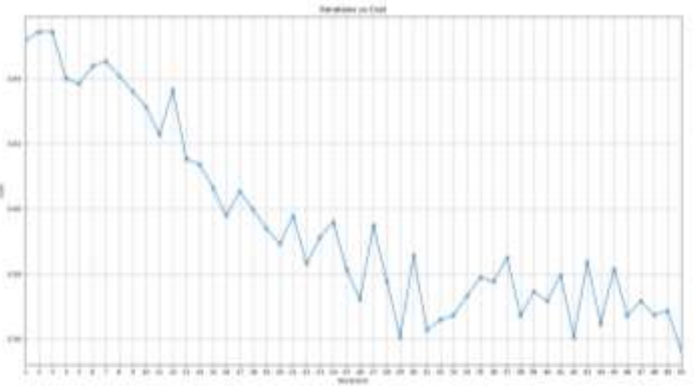


Fig. 6.10: Graph showing the calculated Average Polarity Error at each iteration

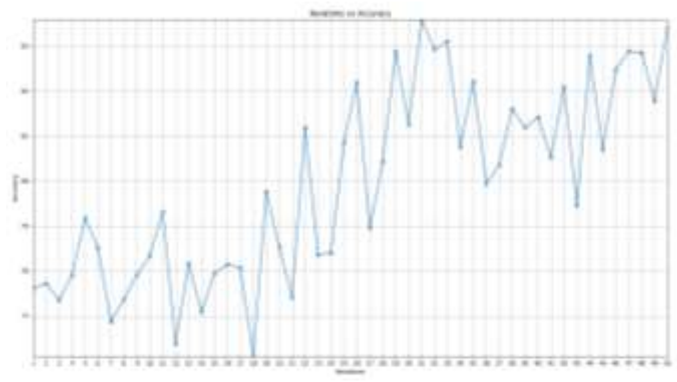


Fig. 6.11: Graph showing the calculated accuracy at each iteration

their weights are trained (a relation with trained weight near zero will not affect the prediction). In the weights training it is seen that some weights moved increased from their initialized weights while others reduced (both for direct and inverse relations). Here the role played by the weights is that they increase or decrease polarities with a percentage indicated by their magnitude. For example, if words A and B are having weight of ‘IsA’ equal to 0.1, then if polarity of A is Pa then polarity of B, Pb = 1.1*Pa It was noticed (shown in Fig. 6.10) that some weights increased or decreased rapidly while others steadily. On increasing the number of iterations from 21 until they converge (which means on an average the weights of corresponding relations do not increase) more accurate results can be observed. Although our results were taken by setting the depth as 2 for computational simplicity depth can be extended to say 10 and thus form a dictionary containing more than 80,000 words. The Average Polarity Error in the above graph is defined by the formula Average Polarity Error = Difference between the classified sentiment value and the reference sentiment value for all the classified words The Average Polarity Error starts decreasing with the increasing number of iterations. At the first iteration, the Average Polarity Error is around 0.48 which starts decreasing and at the 50th iteration , it becomes 0.415 with a little bit of fluctuation in the between . Accuracy = Correctly classified words / Total classified words (correct + incorrect classifications) The mean accuracy (shown in Fig. 6.11) in the whole process increases with respect to the initial accuracy with initial weights.

6.3. Summary

In this chapter, the authors built a lexical resource a domain-specific sentiment lexicons. The methodology leverages the domain-related corpora, word embeddings, ConceptNet and prior sentiment knowledge bases. Following the similar approach, lexicons can also be generated for agriculture, healthcare incorporating depression detection, Hate speech detection, pandemic support, pharmacy, disease detection & treatment, and vaccine performance evaluation, which are not existing right now. It would be interesting to see the implementation and performance of proposed method for lexicon generation in these specific areas. In future, it would also be interesting to work on lexicon generation for low-resource code mixed languages such as Hindi-English, Punjabi-English etc. An interesting future research direction may be the investigation of slang terms, metaphors and sarcasm in the text to assign word polarity more accurately. Handling word sense disambiguation in languages is another area that can be done in future for better performance.

Chapter 7

Text Summarization and Key-Phrase Extraction using Game Theory with Word Embeddings

Game theory has proven to have predictive powers in collective decision-making scenarios and can perform consistent labelling of the data. It was first initiated by Von Neumann and Morgenstern in 1994 as a mathematical framework to model the fundamentals of decision-making in like situations. Game-theoretic frameworks have since been used in various ways to gain knowledge on the use of language and evolution [109]. In literature various approaches utilized WordNet to find out the relationship between words [110]. But WordNet has very limited coverage of words and does not involve word context. Unlike traditional approaches based on WordNet, word embeddings provide semantic and contextual-based similarity and incorporate large and real-world words. This is the main reason to combine word embeddings to game theory approach. This chapter discusses Text summarization and keyphrase extraction application areas of NLP using Game theory

7.1 Evolutionary Game Theory

Evolutionary Game Theory was first brought to light by Smith and Price in 1973. Evolutionary game theory induces an *inductive learning* process wherein a set of entities repeatedly play games with other entities (called neighbors). The players update their understanding of the shape of the game at each iteration. Further strategies are developed according to what has proven to be fruitful previously. The strategy space of a player *i* can be defined as a mixed strategy profile $x_i = (x_1, \dots, x_m)$, where *m* is the count of pure strategies and x_h denotes the probability of player *i* choosing the *h*th pure strategy. In every iteration, a player updates its strategy space in accordance with the payoffs obtained as a result of the games. It then assigns a greater probability to the strategy which has gained higher payoff and the process is continued until a point of equilibrium is achieved – Nash Equilibrium [108,111].

7.1.1 Nash Equilibrium

In a non-cooperative game involving two or more players, Nash Equilibrium—so named after the mathematician John Forbes Nash Jr.—is a suggested solution where each player is assumed to be aware of the equilibrium strategies of the other players and no player stands to gain by changing only their own strategy [112,113]. Each player has a plan they want to use, and according to game theory, no person can gain anything by changing their strategy while the others' stay the same. This combination of tactics and their associated rewards form a Nash equilibrium.
we have used the discrete-time version of the replicator dynamics equation to conduct the experiments in this work:

$$x^h(t+1) = x^h(t)\frac{u(e^h, x)}{u(x, x)} \forall h \in S \tag{7.1}$$

where at each quantum of time t, the players update their strategy space observing the developing environment until equilibrium is attained.

7.2.Proposed Work: Automatic Text Summarization Using Game Theory and Word Embeddings

This section describes the method for automatic extractive text summarization based on game theory and word embeddings. Each player $s \in T$ that takes part in the games is a particular sentence – *s* of the text – *T*. Each strategy of a sentence *s* corresponds to a particular word sense of the words of that sentence. Initially, each player *s* has a uniform normally distributed strategy profile, $S_i = \{1, \ldots, c\}$, where *c* is the total number of strategies corresponding to that player, expressing the likelihood of the player to choose its respective strategies. This profile is keep updating during the course of the games until it converges to the optimal distribution. The games are played using the pay-off matrices and the sentence-similarity matrices of the two sentences, which are described later. Scores are assigned to the sentences using the results obtained from these games. These scores rank the sentences in decreasing order of their importance. The stepwise implementation of the algorithm is described below:

Step 1: Extract a list of all the tokens (sentences) from the text (document).
Step 2: Compute the sentence-similarity matrix using word embeddings, which contains pairwise similarities between all the sentences and depicts each sentences' interaction with every other sentence (neighbour).
Step 3: Compute the sense-similarity matrix between each pair of senses extracted from each word of every sentence. This will further be required to extract the payoff matrices.
Step 4: Construct a strategy profile for each sentence and assign it a probability distribution i.e., distribute a set of probabilities over all the viable strategies (senses) for each sentence (player).
Step 5: Start the games using the mathematical equations of replicator dynamics. Every player is required to play a game with every other player in each iteration. Continue the games until Nash Equilibrium is achieved.
Step 6: The strategy profile of each player is updated as a result of these games. Use these updated profiles to assign scores to the sentences.
Step 7: Rank the sentences according to these scores to form the desired summary.
Pre-processing step divides the given text into tokens. Each token consists of a sentence from the text. All such tokens are saved into a list. The list of sentences is denoted by *list* and the total number of sentences is given by n. Using *list*, we compute an $n \times n$ similarity matrix – *sentence_matrix,* where every element w_{ij} denotes the similarity value between the sentences i and j. Proximity relations are taken into account by increasing the similarity value between two neighbouring sentences (can be checked using bi-gram) by a factor of the mean value of the *sentence_matrix*, as depicted in Algorithm 7.1.
Similarity value between a pair of sentences is calculated using the following approach – for each word of a sentence, we calculate its similarity with every word of the other sentence using various word embeddings such as word2vec, gloVe, fastText, ELMo & BERT and record the highest obtained value for each. The average of all these values gives us the similarity value between the two sentences. Comparison between these embeddings in game theory for text summarization is discussed in the result section. The most popular, cosine similarity is used in Word2Vec word embedding models.

ALGORITHM 7.1: Sentence Similarity Matrix

Input: List *list* = (S_1, S_2, S_3, ... , S_i, ... , S_n) extracted from the text *T*

Output: A 2-D matrix (number_of_sentences × number_of_sentences) denoting the similarity

```
1.  Let n = |list|, average = 0
2.  Let sentence_matrix = 1 for all n * n values
3.  for i = 1 to n do
4.      for j = 1 to n do
5.          Let score = 0, count = 0
6.          if (S_i is not equal to S_j)     // similarity between the same pair of
            sentences are ignored
7.              sentence1 = tokenize the sentence S_i and tag its words
                accordingly
8.              sentence2 = tokenize the sentence S_j and tag its words
                accordingly
9.              for each word1 in sentence1 do
10.                 best_score = 0
11.                 Let similarity_value_list = None
12.                 for each word2 in sentence2 do
13.                     val = similarity-measure (word1, word2) using word
                        embeddings
14.                     similarity_value_list. append(val)
15.                 for k = 1 to | similarity_value_list | do    // filter out the
                    'None' values
16.                     if (similarity_value_list [k] is equal to 'None')
17.                         similarity_value_list [k] = 0.0
18.                     best_score = Extract the maximum value from the
                        similarity_value_list
19.                     Add best_score to score
20.                     Increment count by 1
21.             Divide score by count
22.             set sentence_matrix[i][j] = score
23.             add sentence_matrix[i][j] to average
24. Divide average by n * n
25. for i = 1 to n do
26.     for j = 1 to n do
27.         if (S_i is adjacent to S_j in list)
28.             add average to sentence_matrix[i][j]
29. Return sentence_matrix
```

A list of unique senses (synsets) for every word of the text is extracted using a lexical database (WordNet 3.0 in this case) of the corresponding natural language. Let c denote the total number of available senses corresponding to all the words of the text. The payoff matrices are extracted from a $c \times c$ similarity matrix – *sense_matrix* constructed using the pairwise similarities of all the available senses. The reference knowledge base used in this work is WordNet 3.0 [114].

For each synset, we calculate its similarity with all other synsets. Again, to calculate the similarity measure between senses of words above mentioned word embeddings are used. Algorithm 7.2 gives a more computational and mathematical view for the same.

The partial payoff matrix – Z_{ij} for each game between the players i and j can be obtained by extracting a sub-matrix from the sense similarity matrix (*sense_matrix*) corresponding to the entries corresponding to the senses of the two players. The $m_i \times m_j$ sub-matrix, thus obtained, is called the *payoff_matrix* for the game between players i and j, where m_i and m_j are the count of the senses of players i and j respectively. The greater the similarity value between the senses of the two words, greater is the motivation for the word to take that sense, and hence, utilize the strategy linked to that sense.

ALGORITHM 7.2: Sense Similarity Matrix

```
Input:   List list = ($S_1, S_2, S_3, \ldots, S_i, \ldots, S_n$) extracted from the text T
Output: A 2-D matrix (number_of_synsets × number_of_synsets) denoting the pairwise
 similarities between all the synsets of all the words of the text – sense_matrix
1.  Let all_synsets = None
2.  for each sentence $S_i$ in list do
3.      Tokenize the sentence and tag its words accordingly
4.      W = Extract all the words from sentence that are present in a lexical database W =
        ($w_1, w_2, w_3, \ldots, w_i, \ldots, w_n$)
5.      for each word in W do
6.          Extract all the unique synsets corresponding to word from the same lexical
            database – word-synsets
7.          for synset in word-synsets do
8.              all_synsets.append (synset)
9.  Let len = |all_synsets|, sense_matrix = 0 for all len * len values
    // all_synsets = ($synset_1, synset_2, synset_3, \ldots, synset_i, \ldots, synset_j, \ldots, synset_{len}$)
10. for i = 1 to len do
11.     for j = 1 to len do
12.         Set sense_matrix[i][j] = similarity-measure ($synset_i, synset_j$)
13. Return sense_matrix
```

The sense inventories of the words of the text are used to create the strategy space for the games. A list $C = (1, \ldots, c)$ comprising of all the unique senses is created marking the total space for the games. This strategy space S for the games can be defined in the following manner:

$$\begin{matrix} s_{i1} & s_{i2} & \ldots & s_{ic} \\ . & . & & . \\ . & . & \ldots & . \\ . & . & & . \\ s_{n1} & s_{n2} & \ldots & s_{nc} \end{matrix}$$

Each row corresponds to the strategy space of one particular player and each column corresponds to one particular sense. The components s_{ih} denote the probability with which a player i decides to move ahead with its hth pure strategy over all its other strategies i.e. the probability that the sentence s_i (player) chooses the sense s_h (strategy). Algorithm 7.3 describes the same.

The initial distribution of the mixed strategy space can be done either uniformly (normal distribution as in unsupervised learning) or can extract information from a sense-tagged corpus with pre-labelled values (supervised learning).

Normal distribution follows the following approach:

$$s_{ij} = \begin{cases} |M_i|^{-1}, if\ sense\ s_{ij} \in M_i \\ 0, otherwise \end{cases} \tag{7.2}$$

ALGORITHM 7.3: Strategy Space

Input: List *list* = ($S_1, S_2, S_3, \ldots, S_i, \ldots, S_n$) extracted from the text T

Output: A 2-D matrix (number_of_words x number_of_synsets) denoting the strategy profile of each sentence against all the available senses – *strategy_space*

```
1. Let $strategy\_space$ = 0 for all $n * len$ values
2. for $i$ = 1 to $n$ do
```

```
3.      for $j$ = 1 to *len* do  // Assigning a Normal Distribution to our Strategy Space
4.          strategy_space[$i$][$j$] = $|M_i|^{-1}$ if sense $j$ comes from the sentence $i$
                where $|M_i|$ is the total sense-count of sentence $i$
                                     0   otherwise
5.  Return strategy_space
```

The system dynamics can now be started. Algorithm 7.4 gives a better understanding of the mathematics involved in the process. In each iteration, a player plays a game with each of its neighbors one by one. The payoff for the *h*th strategy is calculated as:

$$u_i(e^h, x) = \sum_{j \epsilon N_i} \left(w_{ij} Z_{ij} x_j\right)_h \tag{7.3}$$

The player's payoff for the iteration is calculated as:

$$u_i(x) = \sum_{j \epsilon N_i} x_i^T \left(w_{ij} Z_{ij} x_j\right) \tag{7.4}$$

Using this method, we can weigh the impact that each sentence has on the options that a candidate sentence has to consider to select its own meaning. The obtained payoff for a sentence i takes into account its own similarity with the sentence $j - w_{ij}$; the similarities among the senses of its words and the words of the sentence $j - Z_{ij}$, and the sense predilection of sentence $j - x_j$. Strategies pertaining to higher payoffs emerge at every iteration as a result of the selection criterion. After the process terminates, each player chooses its sense according to these restrictions.

The system may run for a pre-defined number of iterations or until convergence, as achieved by the following equation of Nash Equilibrium:

$$\Phi_i = arg \max_{h=1,\ldots,c} x_{ih} \tag{7.5}$$

The result of the replicator dynamics assigns to each word of each sentence the sense that the word is most likely to assume in the given context. The top few senses of each word are taken to form the overall meaning of the sentence.

ALGORITHM 7.4: Replicator Dynamics

```
Input: list, sentence_matrix, sense_matrix, strategy_space
Output: strategy_space gets updated
1.  Let number_of_iterations = 100                    // Assumption
2.  for $i$ = 1 to number_of_iterations do
3.      for $j$ = 1 to $n$ do                          // Player
4.          for $k$ = 1 to $n$ do                      // Neighbour
5.              Let payoff_matrix = a (sense_count_of_player x
                sense_count_of_neighbour)
                submatrix extracted accordingly from the sense_matrix
6.              The payoff for each strategy (current_payoff) is calculated according
                to the following equation: sentence_matrix[j][k] * dot-product
                (payoff_matrix, strategy_space[k])
7.              The strategy_payoff is calculated as Σ current_payoff
8.          The player_payoff is calculated as Σ dot-product (current_payoff,
            strategy_space[j]$^T$)
9.          Let updation_values for a player j = strategy_payoff / player_payoff
10.         Update strategy_space[j] according to the obtained updation_values
```

Now that we have the updated strategy profiles of the players, we know the optimal senses corresponding to each sentence. For each sentence, we compare its top senses (strategies) with every sense (strategy) of the other sentence to find the similarity between the two. Using the mean of the obtained data, we assign each sentence a similarity score as depicted in Algorithm 7.5. The sentence which has the maximum similarity is given the highest score. A descending order of the scores gives the final ranking of the sentences for the summary.

ALGORITHM 7.5: Assignment of Scores

```
Input: List list = ($S_1$, $S_2$, $S_3$, ... , $S_i$, ... , $S_n$) extracted from the text T, strategy_space
Output: An output array scores containing the similarity scores of each sentence
    Let sentence_synsets = None
    for each sentence in T do
        important_senses = Extract a list of the top important senses (synsets) using
        the strategy_space matrix
        sentence_synsets.append (important_senses)
    Let scores = []
    for each sentence $S_i$ in T do
        val = 0
        for each sentence $S_j$ in T do
            for each sense $s_i$ in sentence_synsets[i] do
                for each sense $s_j$ in sentence_synsets[j] do
                    add similarity_measure (sense $S_i$, sense $S_j$) to val
        val = val / (|sentence_synsets|*|sentence_synsets|)
        scores. append (val)
    Return scores
```

7.2.1 Illustration through an Example

This section discusses an example to illustrate the algorithm proposed in this work. The following document has been taken as input to the algorithm:

A smart home speaker was the need of the hour. I was looking for the same on the Internet when I stumbled upon the perfect recommendation - the Echo Dot 3rd generation from Amazon. I immediately decided to move forward with it. The package containing the product was delivered to me within two days of its purchase. After using the product, I realised that the new Echo Dot from Amazon is the best compact smart speaker in the market by a wide margin. The quality of sound is extraordinary. Alexa has become smarter than ever with reasonable introduction to smart home control. The microphone sensitivity has also been improved significantly. However, I do feel *that over a period of time the outer fabric cover might get dusty and the new power port is not as universal as it could have been.*

The document is a review of the "Echo Dot 3rd Generation" from Amazon. The following steps illustrate the procedure to be followed to rank the sentences in order to produce a summary.

First step requires to divide the input text into nine sentences – S0, S1, S2, S3, S4, S5, S6, S7 and S8. We then proceed to calculate the sentence similarity matrix as illustrated in *Algorithm 7.1*. The sentence similarity matrix obtained for this example is shown in *Table 7.1*.

Table 7.1: Sentence Similarity Matrix

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	0.000	0.588	0.333	0.291	0.554	0.310	0.336	0.265	0.335
1	0.541	0.000	0.624	0.254	0.540	0.259	0.166	0.228	0.258
2	0.333	0.624	0.000	0.624	0.333	0.333	0.333	0.333	0.333
3	0.268	0.275	0.532	0.000	0.685	0.270	0.164	0.254	0.320
4	0.392	0.510	0.305	0.619	0.000	0.541	0.167	0.218	0.352
5	0.447	0.444	0.333	0.430	0.735	0.000	0.560	0.400	0.555
6	0.334	0.178	0.261	0.185	0.187	0.506	0.000	0.571	0.375
7	0.350	0.356	0.305	0.358	0.358	0.353	0.639	0.000	0.785
8	0.351	0.340	0.333	0.374	0.419	0.369	0.315	0.675	0.000

Algorithm 7.2 states the procedure to calculate the sense similarity matrix using word embeddings (Word2Vec in this illustration). The output obtained after this step is depicted in *Table 7.2*.

Table 7.2: Sense Similarity Matrix *77 rows X 77 columns*

	0	**1**	**2**	**3**	**4**	**5**	**6**	**...**	**75**	**76**	**77**
0	1.000	0.235	0.117	0.133	0.285	0.461	0.307	...	0.133	0.285	0.190
1	0.235	1.000	0.111	0.125	0.266	0.285	0.285	...	0.125	0.266	0.272
2	0.117	0.111	1.000	0.333	0.133	0.142	0.266	...	0.375	0.133	0.090
3	0.133	0.125	0.333	1.000	0.153	0.166	0.307	...	0.470	0.153	0.100
4	0.285	0.266	0.133	0.153	1.000	0.363	0.363	...	0.153	0.500	0.210
5	0.461	0.287	0.142	0.166	0.363	1.000	0.400	...	0.166	0.363	0.222
6	0.307	0.285	0.266	0.307	0.363	0.400	1.000	...	0.307	0.363	0.222
7	0.235	0.333	0.111	0.125	0.266	0.285	0.285	...	0.125	0.266	0.454
8	0.111	0.105	0.315	0.400	0.125	0.133	0.250	...	0.588	0.125	0.086
9	0.153	0.142	0.142	0.153	0.181	0.200	0.200	...	0.166	0.181	0.111
10	0.250	0.235	0.117	0.133	0.285	0.307	0.461	...	0.133	0.285	0.190
...	...	...	...	...	...	...	...	...	...	...	...
74	0.285	0.266	0.133	0.153	0.500	0.363	0.363	...	0.153	0.666	0.210
75	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000
76	0.153	0.142	0.142	0.153	0.181	0.200	0.200	...	0.166	0.181	0.111
77	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000

After this step we initialize our strategy space according to *Algorithm 7.3* and assign normal probabilities to all the strategies (senses) corresponding to each player (sentence). We then proceed to perform Replicator Dynamics (*Algorithm 7.4*) as we initiate the games between the sentences. The scores assigned to each strategy (sense) of every sentence is shown in *Table 7.3*

Table 7.3: Strategy Space (After Replicator Dynamics) *9 rows X 77 columns*

	0	**1**	**2**	**3**	**4**	**...**	**75**	**76**	**77**
0	9.881	9.881	4.940	0.494	1.000	...	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000
3	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000
4	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000
5	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000
6	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000
7	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	...	0.000	0.000	9.881

Table 7.4: Important strategies (senses) corresponding to the sentences

	0	1	2	3	4
0	need.n.01	hour.n.01	angstrom.n.01	smart.n.01	home.n.01
1	iodine.n.01	perfective.n.01	echo.n.01	point.n.09	coevals.n.01
2	iodine.n.01	angstrom.n.01	smart.n.01	home.n.01	speaker.n.01
3	package.n.01	two.n.0	days.n.01	purchase.n.01	angstrom.n.01
4	iodine.n.01	echo.n.01	point.n.09	market.n.01	margin.n. 01
5	quality.n.01	sound.n.01	angstrom.n.01	smart.n.01	home.n.01
6	control.n.01	introduction.n.01	smart.n.01	angstrom.n.01	smart.n.01
7	sensitivity.n.01	introduction.n.01	better.v.02	angstrom.n.01	smart.n.01
8	iodine.n.01	power.n.01	feel.n.01	time_period.n.01	time.n.01

Table 7.5: Score of the sentences

Sentence	S0	S1	S2	S3	S4	S5	S6	S7	S8
Score	2.490	2.505	2.311	2.481	2.535	2.567	2.566	2.520	2.640
Rank	(7)	(6)	(9)	(8)	(4)	(2)	(3)	(5)	(1)

We now have the important strategies (senses) corresponding to every sentence in the text – *Table 7.4*. The score of every sentence is impacted by the very sense that is categorized as important for it. The scores thus assigned are shown in *Table 7.5*. After the assignment of scores, a summary of size (the required number of sentences) '*k*' is expected to be generated. The top '*k*' sentences are then extracted on the basis of their scores and are given as the output. Let the required number of sentences for the summary ('*k*') be 5. *Table 7.6* shows a summary of length five.

Table 7.6. Final Summary of the Illustrative Text

Rank	Sentences
1	However, I do feel that over a period of time the outer fabric cover might get dusty and the new power port is not as universal as it could have been.
2	The quality of sound is extraordinary.
3	Alexa has become smarter than ever with reasonable introduction to smart home control.

4	After using the product, I realised that the new Echo Dot from Amazon is the best compact smart speaker in the market by a wide margin.
5	The microphone sensitivity has also been improved significantly.

7.2.2. Experimental Setup, Results and Discussion

This section discusses datasets description, evolutionary metrics, and comparison with state-of-art methodologies.

7.2.2.1 Evaluation Dataset Description and Metrices

For the experiments, we have performed different experiments on DUC 2002 dataset from Document Understanding Conference (DUC). DUC 2002 consist of 567 news articles with 59 number of clusters in the English language.
The proposed work used the *ROUGE* package for the automatic evaluation of algorithm [21, 25]. The toolkit has been assumed by DUC for automatic summarization evaluations. The count of overlapping units like the *n*-gram, the sequence of the words and word pairs between the candidate summary and the reference summary give a measure of the quality of the summary.

ROUGE-N is an *n*-gram recall measure and is computed as follows:

$$ROUGE - N = \frac{\sum_{S\in\{Ref\sum\}}\sum_{n-gram\in S} Count_{match}(n - gram)}{\sum_{S\in\{Ref\sum\}}\sum_{n-gram\in S} Count(n - gram)} \quad (7.6)$$

where *n* denotes the size of the *n*-gram, $\text{Count}_{\text{match}}$ (*n*-gram) gives the maximum number of *n*-grams co-occurring in a candidate summary and a set of reference summaries and Count (*n*-gram) gives the count of *n*-grams in the reference summaries.

While the older versions of the ROUGE packages used a score based on recall for the evaluation of summaries, the newer versions do the evaluation based on the following three metrics: ROUGE-N *Precision*, ROUGE-N *Recall* and ROUGE-N *F-Score*, where N can take the values 1, 2, 3, 4 etc. Separate scores for each 1, 2, 3, 4 grams and the SU4: skip bigram are reported by the ROUGE toolkit. ROUGE-L examines the longest subsequence between standard and generated summaries. The ROUGE Version 1.5.5 has been used for the evaluation of our system.

7.2.2.2 Comparison with DUC systems on DUC 2002 dataset

This section compares the proposed method with popular five DUC systems on DUC 2002 dataset. Brief description of these five-systems sys code 21, sys 27, sys 28, sys 29 and sys31 is given in Table 7.7. These systems are considered to have the best ROGUE scores and have been widely used for comparisons. Their summaries have been listed on the DUC-NIST website. A comparison-based evaluation of ROGUE-1, ROGUE- 2 and ROGUE-SU4 scores of proposed approach, the baseline and the top DUC mentioned systems is performed. The results are shown in Table 7.8 and Fig. 7.1. It is very clear that the proposed system based on word embedding and Game Theory (TSWE&GT) outperforms the others.

Table 7.7: A brief description of the top five systems participating in DUC 2002

System Code	System ID	System Description
21	Wpdv-xtr.v1	Supervised sentence classification with WPDV (Weighted Probability Distribution Voting)
27	Ntt.duc02	Supervised sentence classification with SVM (Support Vector Machines)
28	Ccsnsa.v2	Supervised sentence extraction with the Hidden Markov Model (HMM) and the LRM (Logistic Regression Model)
29	Kul.2002	Unsupervised sentence extraction + topic segmentation
31	ULeth 131m	Unsupervised sentence extraction + text segmentation

Table 7.8: Results on the DUC 2002 dataset (Stop-words included)

System	ROGUE-1	ROGUE-2	ROGUE-SU4
TSWE&GT (proposed)	**53.4**	**28.1**	**28.9**
System 27	50.1	24.1	25.3
System 31	49.1	23.8	25.1
System 28	48.3	23.2	24.7
System 21	47.9	22.8	24.1
DUC 2002 Baseline (Lead)	47.5	22.6	24.4
System 29	46.7	21.7	23.5

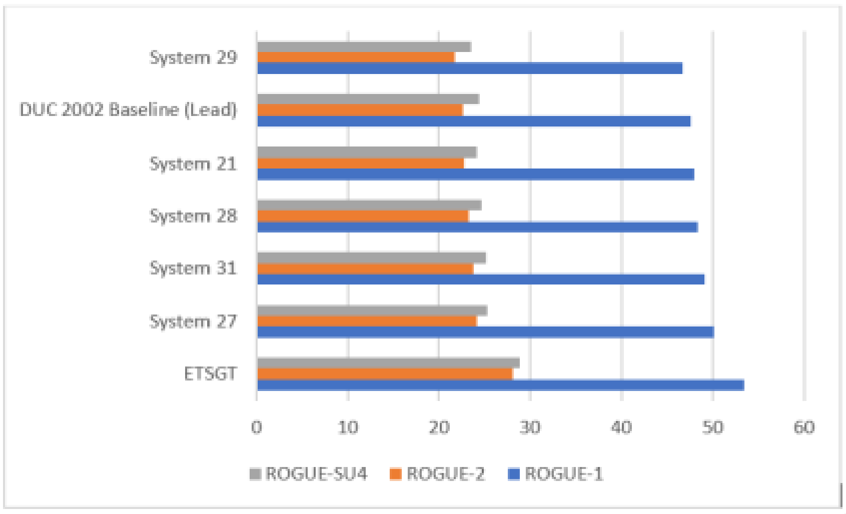


Fig. 7.1. Results for system comparison on DUC 2002 data (stop words included)

7.2.2.3 Performance Comparison on DUC 2002 Dataset

We have performed a comparative evaluation of our proposed system with widely used baseline approaches and state-of-the-art systems published in the literature on DUC 2002, as shown in Table 7.9.

Description of Baseline approaches are as follows:

TextRank [115] a graph-based algorithm depicts a document as a graph of sentences, with the edges connecting two sentences based on how similar they are.

LexRank [116] is to determine the significance of each sentence in the document, LexRank applies the idea of eigenvector centrality to the graph representation of sentences.

LSA (Latent Semantic Analysis) [117]: By applying Singular Value Decomposition (SVD) on a document matrix D of size m x n, consisting of m sentences and n number of terms, LSA (Latent Semantic Analysis), attempts to identify salient sentences in the document.

KLSum [74] :The output summary of the document is produced by the text summarization algorithm KLSum based on the Kullback-Lieber (KL) divergence criterion.

SumBasic [71] is a greedy search approximation technique that combines a component to re-weight the word probabilities with a frequency-based phrase selection component to reduce redundancy.

Integer Linear Programming [118] A phrase-based summarization technique that uses integer linear programming to encapsulate the grammaticality restrictions across phrase dependencies.

The entity graph-based system Egraph [120] represents documents using a bipartite graph of phrase and entity nodes. They created a system called "Egraph + Coh." that has a coherence metric that is determined by utilising the average outdegree of an unweighted projection graph.

Tgraph [121] is a graph-based, unsupervised text summarising system that models topics using Latent Dirichlet Allocation (LDA). Parveen et al"Tgraph .'s + coh" proposal generates a summary by taking into account coherence determined using a weighted projection graph.

URANK [84] is a unified rank methodology built on a graph model that manages multi-document summarization at the same time.

The graph-based unsupervised extractive text summarization method CoRank combines the graph-based ranking model with the word-sentence relationship.

SummerCoder [119]is an unsupervised framework for extractive text summarization based on deep auto-encoders.
GT_AS [122] is a game theory-based approach for text summarization. PE_MSC [121]is a text summarization approach using Entailment based minimum set cover.

In comparison to existing summarization algorithms, our method achieved very good ROUGE scores on the DUC 2002 document extractive text summarization. On this dataset, we obtained a ROUGE-1 score of 53.4, a ROUGE2 score of 28.1, and a ROUGE-SU4 score of 46.7. In terms of ROUGE-1, ROUGE-2, and ROUGE-SU4 scores, proposed framework (TSWE&GT) outperforms the aforementioned baseline and existing state-of-the-art text summarizers on the DUC 2002 dataset, including ILP, graph-based approaches like Tgraph, Egraph, and URANK, and even those that are based on deep auto-encoders like SummerCoder. Table 7.10 shows the ROUGE scores for various word embedding models with game theory on DUC 2002 dataset. Fig. 7.3. Represents the graphical impact of various word embedding models with game theory. The results shows ensemble Bert with game theory method outperforms others.

Table 7.9: ROUGE Scores on DUC 2002 using baseline, State-of-art and TSWE&GT approach

Methods	ROUGE-1	ROUGE-2	ROUGE-SU4
TextRank	47.0	19.5	42.8
LexRank	42.9	21.1	37.1
LSA	43.0	21.3	40.0
KLSum	38.3	16.9	32.2
SumBAsic	39.6	17.3	35.1
ILP	45.4	21.3	42.8
Egraph+coh	47.9	23.8	-
Tgraph + coh	48.1	24.3	-
URANK	48.5	21.5	-
CoRank	52.6	25.8	45.1
SummCoder	51.7	27.5	44.6
KP-centrality	49.8	24.5	43.7

GT_TS by Ahemed	-	35.5	-
PE_MSC	50.0	22.1	22.8
TSWE&GT (proposed)	53.4	28.1	46.7

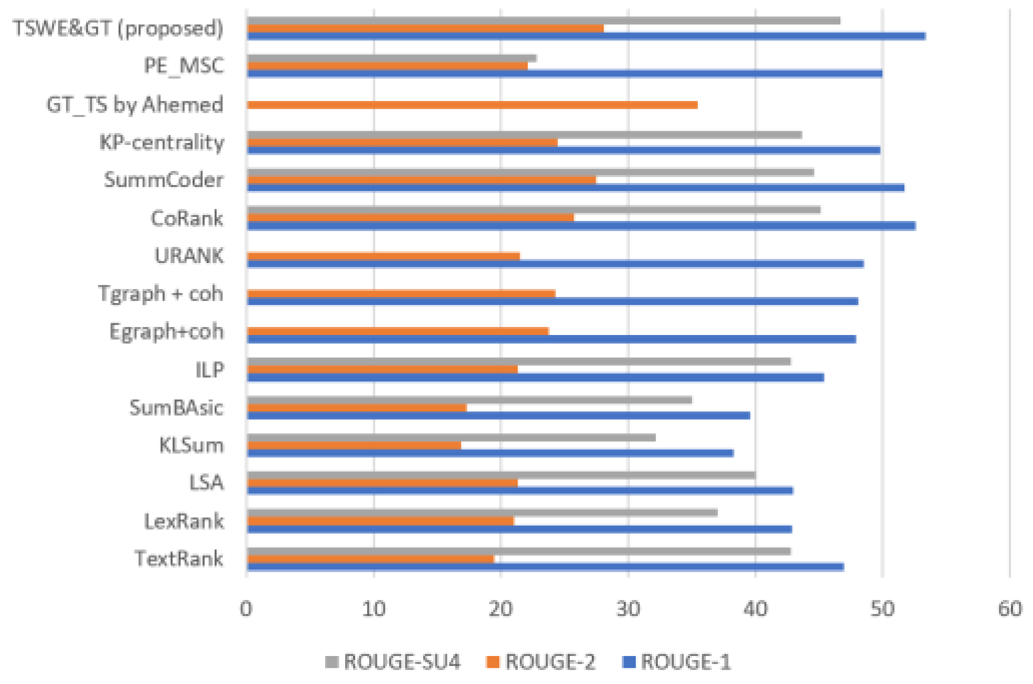


Fig. 7.2. ROUGE scores on DUC 2002 using baseline, state-of-the-art and TSWE&GT approach

Table 7.10: ROUGE scores for various word embedding models on DUC 2002 dataset

Word Embeddings	ROUGE-1	ROUGE-2	ROUGE-SU4
Word2Vec +GT	52.8	26.5	26.3
GloVe +GT	51.9	25.7	25.7
FastText+ GT	52.3	26.2	26.8
BERT+ GT	**53.4**	**28.1**	**28.9**
ELMo+ GT	53.0	27.4	27.2

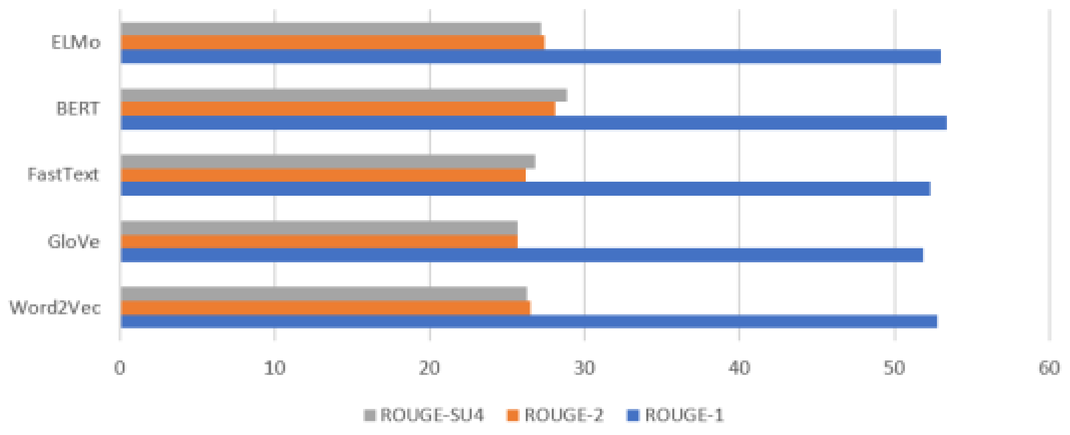


Fig. 7.3. Graphical Representation of results for word embedding models with game theory

7.3 Proposed Automatic Keyphrase Extraction using Fuzzy-Based Evolutionary Game Theory Approach

With the increased usage of social media, a massive amount of textual data is generated, which is used for applications such as commerce trend analysis, opinion mining, information retrieval, and so on. Automatic key extraction is a critical and important operation in this situation. Many graph-based approaches that employ co-occurrence as edge weight have been developed, but these algorithms ignore the semantic relationships between words. This chapter offers an unsupervised Automatic Key Extraction framework that combines evolutionary game theory with a fuzzy method. The suggested methodology treats Automatic Key Extraction as a consistent labelling problem to ensure that candidates are consistently classified as key or non-key. Various datasets are employed for experimentation, and the outcomes suggest that the proposed approach outperforms the state-of-art methods. The overall proposed methodology is divided in 5 steps discussed in forthcoming subsections in detailed.

7.3.1 Candidate key phrase extraction

This segment includes extraction of candidate keyphrases from intext text. The input text is passed through a parser of Spacy to get noun phrase chunks from input text.

noun broken in to tokens. All the noun phrases extracted are considered as candidate keyphrases $CKP = [KP_1, KP_2, KP_3, \dots, KP_i, \dots, KP_n]$

7.3.2 Candidate Keyphrase Fuzzy Similarity Matrix and Sense Fuzzy Similarity Matrix

For Keyphrase Fuzzy Similarity Matrix (KPFSM) input is candidate keyphrase $CKP = [KP_1, KP_2, KP_3, \dots, KP_i, \dots, KP_n]$ extracted in previous step. And output is a 2-D *KPFSM* matrix (number_of_keyphrases x number_of_keyphrases) marking the similarity between every pair of keyphrases. The next matrix is fuzzy sense similarity matrix (*SFSM*) where input is all the senses of $CKP_i \in CKP$ and output is a 2-D matrix (number_of_synsets x number_of_synsets) marking the pairwise fuzzy similarity values of the synsets of the key phrases of *TEXT*

The similarity between two *CKP* are calculated using Fuzzy Jaccard similarity measure. It is a modification of the Jaccard similarity coefficient that takes into account partial matching between two sets. The traditional Jaccard similarity coefficient is defined as the ratio of the size of the intersection of two sets to the size of the union of the sets. However, the fuzzy Jaccard similarity measure considers that elements in the intersection of the sets may not match exactly, but may have a certain degree of similarity.

To calculate the fuzzy Jaccard similarity, a threshold value is set for the degree of similarity required for elements in the intersection to be considered as matches. Then, the degree of similarity between each pair of elements in the sets is calculated using a fuzzy matching algorithm, such as the Jaro-Winkler distance or Levenshtein distance. The pairs of elements with similarity greater than the threshold value are considered as matches and included in the intersection of the sets.

Finally, the fuzzy Jaccard similarity is calculated by dividing the size of the fuzzy intersection of the sets by the size of the fuzzy union of the sets. The fuzzy union of the sets includes all elements from both sets, but removes duplicates and includes only the highest degree of similarity for each element.

7.3.3 Strategy Space Generation

A 2-D matrix (number_of_keyphrases x number_of_synsets) marking the strategy profile of each key phrase against all synsets – *strategy_space*. Where input is all the candidate keyphrases $CKP = [KP_1, KP_2, KP_3, \ldots, KP_i, \ldots, KP_n]$. Nash Equilibrium using Dynamic Replicator and final scores are shown in algorithm 7.6 and algorithm 7.7 respectively.

$$strategy\_space[i][j] = \begin{cases} |count_i|^{-1} & \text{if synset } j \text{ is from the key phrase } i \\ 0 & \text{Otherwise} \end{cases}$$

where $|count_i|$ is the number of synsets of key phrase i

ALGORITHM 7.6: Replicator Dynamics for keyphrase extraction

Input: *list*, keyphrase*_matrix*, *fuzzy sense_matrix*, *strategy_space*

Output*: strategy_space* gets updated

```
1.  Let number_of_iterations = 200                // Assumption
2.  for i = 1 to number_of_iterations do
3.      for j = 1 to n do                          // Player
4.          for k = 1 to n do                      // Neighbour
5.              Let payoff_matrix = a (fuzzy sense_count_of_player x fuzzy
                sense_count_of_neighbour)
                submatrix mined accordingly from the fuzzy sense_matrix
6.              The payoff for each strategy (current_payoff) is calculated according
                to the following equation: fuzzy sentence_matrix[j][k] * dot-product
                of (payoff_matrix, strategy_space[k])
7.              The strategy_payoff is calculated as Σ current_payoff
8.          The player_payoff is calculated as Σ dot-product of (current_payoff,
            strategy_space[j]^T)
9.          Let updation_values for a player j = strategy_payoff / player_payoff
10.         Update strategy_space[j] rendering to the found updation_values
```

ALGORITHM 7.7: Scores for final keyphrases in result

```
Input: tokens = [$KP_1$, $KP_2$, $KP_3$, ... , $KP_i$, ... , $KP_n$], strategy_space
Output: A list – final_scores containing the computed scores of each key phrase
    Let keyphrase_synsets = None
    for each keyphrase in Text do
        important_senses = Most important synsets from the strategy_space matrix
        keyphrase_synsets.append (important_senses)
    Let final_scores = []
    for each keyphrase $KP_i$ in Text do
        value = 0
        for each keyphrase $KP_i$ in Text do
            for each sense $s_i$ in keyphrase _synsets[i] do
                for each sense $s_j$ in keyphrase _synsets[j] do
                    add fuzzy similarity_measure ($synset_i$, $synset_j$) to value
        value = value / (|keyphrase_synsets|*| keyphrase _synsets|)
        Final_scores. append (value)
    Return final_scores
```

7.3.4. EXPERIMENTAL SETUP AND RESULT DISCUSSION

For a comprehensive evaluation of the proposed method, two scientific publication datasets are used. The dataset is comprised of the corresponding author manually labelled key-phrases (gold standard), the research paper titles and abstracts. Both the datasets comprises of research papers from ACM Conference on Data Mining (KDD) and Knowledge Discovery and ACM World Wide Web (WWW).

Table 7.11: Statistics of the Datasets

Dataset	#Abs/#KPs (All)	MissingKPs (%)	#Abs/#KPs (Loc.)	AvgKPs	#uni.	#bi	#tri	#> trigrams
KDD	365/1471	51.12	315/719	4.03	363	853	189	66
WWW	425/2073	56.39	388/904	4.87	680	1036	247	110

Table 1 presents statistics for two datasets, including the total number of key-phrases and abstracts, the percentage of key-phrases absent from the abstract, the number of key-phrases located, and the mean number of key-phrases per paper. The table also shows the distribution of key-phrases with one, two, three, or more tokens. Three common features are shared by both datasets, namely, an average of four to five key-phrases per paper, approximately half of the key-phrases not being present in the abstract, and a minority of key-phrases being 3-grams or more. These statistical features provide insights into the challenges of extracting key-value pairs from these datasets.

To evaluate the results of key-extraction, precision, recall, F1-score, and Mean Reciprocal Rank (MRR) are commonly used, and practically all prior works have done the same. As a result, to ensure consistency when compared to other state-of-the-art algorithms, we maintained our evaluation metric constant.

where kc is the number of successfully extracted key-value pairs, ke is the total number of extracted key-value pairs, and ks is the total number of author-labeled standard key-value pairs.

MRR is used to determine how each document's first accurate key is rated. MRR is defined for a document d as

$$MRR = (1/|TD|)*\text{summation}_{k \in TD} 1/\text{rank}_K \qquad (7.7)$$

where TD is the set of target documents and rankK denotes the rank of the first right key-word among all extracted key-words. We perform experiments on the best (top) n (n = 2 and 8) extracted key-phrases for the assessment scores in our studies. For the purposes of comparison, we used Porter Stemmer in our investigation, which assisted in the reduction of both expected and gold key-values to a base form. Tables 7.12 and 7.13 compare the results of our system KPFEGT with those of other cutting-edge systems on WWW and KDD datasets at top n predicted key-phrases, where n spans from 2 to 8 . We can deduce from the tables below that the overall findings of the KDD dataset are better than those of the WWW dataset. KPFEGT outperforms all comparative techniques on both the KDD and WWW datasets, as demonstrated in Tables 7.12 and 7.13. For example, on both datasets, at top n = 8 predicted key-phrases, KPEGT outperforms all other performance measures.

Table 7.12: Comparison of proposed KPFEGT with other state-of-the-art Approaches (with top n = 2)

Method	KDD	KDD	KDD	KDD	WWW	WWW	WWW	WWW
	P	R	F1	MRR	P	R	F1	MRR
TF-IDF	0.175	0.087	0.116	0.289	0.183	0.075	0.106	0.275
TextRank [4]	0.145	0.721	0.096	0.221	0.150	0.061	0.087	0.222
SingleTPR	0.182	0.090	0.120	0.287	0.168	0.069	0.097	0.275
Position Rank [6]	0.172	0.085	0.114	0.280	0.162	0.066	0.094	0.249
FKE [1]	0.191	0.095	0.127	0.309	0.210	0.086	0.122	0.316
GTKPE [2]	0.228	0.113	0.151	0.363	0.221	0.090	0.128	0.336
KPFEGT	0.254	0.20	0.224	0.434	0.255	0.135	0.177	0.453

Table 7.13: Comparison of KPFEGT with other state of the art approaches (with top n = 8)

Method	KDD	KDD	KDD	KDD	WWW	WWW	WWW	WWW
	P	R	F1	MRR	P	R	F1	MRR
TF-IDF	0.091	0.178	0.121	0.336	0.095	0.154	0.118	0.321
TextRank [4]	0.075	0.147	0.100	0.270	0.081	0.132	0.101	0.273
SingleTPR	0.088	0.172	0.117	0.329	0.090	0.145	0.111	0.326
Postion Rank [6]	0.085	0.166	0.113	0.324	0.089	0.144	0.110	0.303
FKE [1]	0.094	0.182	0.124	0.358	0.098	0.159	0.122	0.361
GTKPE [2]	0.098	0.191	0.130	0.392	0.102	0.165	0.126	0.378
KPFEFGT*	0.132	0.212	0.163	0.43	0.167	0.23	0.194	0.421

Summary

This chapter discussed the two important natural lamguage processing application area text summarization and keyphtase extraction using gam etheory and word embeddings. For text summarization hybrid approach using evolutionary game theory and various word embedding models is capable to incorporate the semantic of input text and preserve the cohesiveness of the resultant summary. It is found that the proposed method successfully ranks each sentence in an order of importance by finding the mutual similarity of that sentence with all other sentences. The performance of generated summaries has been evaluated using ROUGE score on DUC 2002 dataset.

Several experiments are performed and it has been found that proposed technique outperformed the state-of-art techniques on ROUGE-1, ROUGE-1 and ROUGE-SU4.

For Kephrase extraxtion, The proposed method defined keyphrases and developed a logical framework to represent them mathematically as payoffs in keyphrase games. The work evaluated proposed method on two widely used scientific publication datasets and found that it outperforms most existing systems, demonstrating the potential of this approach.

Chapter 8

Conclusion and Future Work

In this chapter, we conclude our work presented in the thesis in section 8.1. In section 8.2, we present the contributions made by the research work carried out in the field of Natural Language Processing. The findings and the interpretation of the results are given in section 8.3. In section 8.4, we state the possible future extensions of our proposed algorithms.

8.1 Conclusion

In this thesis, we made an attempt to solve the various tasks of Natural language processing using soft computing techniques. Work presented in this thesis successfully demonstrates that the fuzzy logic in application areas of natural language processing can make it close to real life by handling the uncertainties of real life. We also argue that there is a need to automatically assign membership values for fuzzy semantic relationships between words of Fuzzy Hindi WordNet and also proposed an approach for this reassignment. We demonstrated that fuzzy Interval-valued fuzzy Hindi WordNet gives better results than Type-1 fuzzy Hindi wordnet. We also concluded that domain-specific lexicons are better in the field of sentiment analysis. In the thesis, it is shown that the game-theoretic approach worked better than centrality measures for NLP applications.

8.2 Contribution

This thesis contributes to the knowledge in the area of NLP. Specifically, it introduces new algorithms for various NLP applications using soft computing. The significant contributions of this thesis are:

1. We designed a new algorithm for Hindi-English Social Media Text correction using fuzzy graph connectivity measures and word embeddings
2. We fuzzified the crisp semantic relations of Fuzzy Hindi WordNet to improve the results of WSD.
3. We automatically constructed interval-valued fuzzy Hindi wordnet using lexico-syntactic patterns and assigned membership values to IVF fuzzy semantic relations by using embeddings.
4. We developed domain-specific sentiment lexicons for sentiment analysis task.
5. We performed Text summarization and keyphrase extraction using the Game Theoretic approach.

8.3 Findings

1. Effective Text normalization can improve the performance of NLP algorithms.
2. Results of WSD of text improved by fuzzifying the crisp semantic relations of Fuzzy WordNet.
3. Interval-valued fuzzy Hindi WordNet provides better results than crisp Hindi wordnet and Type-1fuzzy Hindi WordNet.
4. Domain-specific lexicons are better than classical generic lexicons for sentiment analysis.
5. Game theory in NLP provides better results than graph centrality measures in NLP

8.4 Future Work

The research work carried out and presented in the thesis is a humble attempt for NLP solutions using soft computing. Thus, there is high scope for further improvement for research as follows:

1. For text normalization, the work can be extended for different code-mixed languages. In social media text, Devanagari or English is also mixed with other scripts,

2. In future, Type II fuzzy can also be the part of the correction system to handle the uncertainties of type 1 fuzzy, and it would be interesting to analyse the results.
3. WordNet exists for several languages, for instance, IndoWordNet (for Indian languages), EuroWordNet etc., establishing and evaluating the performance of extending WordNet to Interval-Valued WordNet with respect to these resources is an intriguing future direction.
4. For many lexicons such as sentiwordnet, WordNet is used as a base, so the extension of sentiwordnet to Interval-Valued Fuzzy sentiwordnet can improve resource performance.
5. In future, it would also be interesting to work on lexicon generation for low-resource code mixed languages such as Hindi-English, Punjabi-English etc.
6. An interesting future research direction may be the investigation of slang terms, metaphors and sarcasm in the text to assign word polarity more accurately.
7. In the future Game theory can be explored for other NLP areas.

LIST OF PUBLICATIONS

Journals

1. Minni Jain, Rajni Jindal, Amita Jain. “Code-Mixed Hi-En Text Correction using Fuzzy Graph and Word Embedding” Expert Systems Journal, Wiley, SCIE. IF-3.3 DOI: _https://doi.org/10.1111/exsy.13328

2. Minni Jain, Rajni Jindal, Amita Jain “Lexical Semantics Identification using Fuzzy Centrality Measures and BERT Embedding” National Academy Science Letters, Springer, SCIE, IF- 1.1 DOI: https://doi.org/10.1007/s40009-023-01310-2

3. Minni Jain, Rajni Jindal, Amita Jain “Automatic Construction of Interval-Valued Fuzzy Hindi WordNet using Lexico-Syntactic Patterns and Word Embeddings” ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP). (Major Revision Submitted)

4. Minni Jain, Rajni Jindal, Amita Jain “DoSLex: Domain-Specific Lexicon for Unsupervised Sentiment Analysis of Low–Resource Languages” Language Resources and Evaluation, Springer. (Communicated)

5. Minni Jain, Rajni Jindal, Amita Jain “Extractive Text Summarization using Game Theory with Word embeddings” Multimedia Tools & Applications, Springer, SCI. (Communicated)

Conferences:

1. Minni Jain, Rajni Jindal, Amita Jain “Building Domain-Specific Sentiment Lexicon using Random walk-based model on Common-sense Semantic Network” 6th International Conference on Innovative Computing And Communication (ICICC 2023), Delhi (Presented).

2. Minni Jain, Rajni Jindal, Amita Jain “Building Domain-Specific Sentiment Lexicon using Label Propagation” SCIS&ISIS 2018, Joint 10th International

Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS), Toyama, Japan, December 5-8, 2018. IEEE 2018, ISBN 978-1-5386-2633-7, Japan. (Presented)

3. Minni Jain, Rajni Jindal, Amita Jain “Keyphrase extraction using fuzzy based game theory” , 4th International Conference on Data Analytics & Management (ICDAM-2023), Organized By: London Metropolitan University, London, UK (Venue Partner) in association with Karkonosze University of Applied Sciences, Jelenia Gora, Poland, Europe & Politécnico de Portalegre, Portugal, Europe &BPIT, GGSIPU, Delhi Date: 23rd - 24th June, 2023, London (Presented).

[1] LA Zadeh, "Fuzzy Sets. Information and Control", 8(3):338-353, 1965.

[2] A. Rosenfeld, LA Zadeh, Fu KS, K. Tanaka, M. Shimura, "Fuzzy Sets and their Application to Cognitive and Decision Process", Eds. New York: Academic, 77–97, 1975.

[3]S. Mathew, and M. S. Sunitha. "Types of arcs in a fuzzy graph." Information sciences 179, no. 11: 1760-1768, 2009.

[4]P. Bhattacharya. "Some remarks on fuzzy graphs." Pattern recognition letters 6, no. 5: 297-302, 1987

[5]A. Jain, and D. K. Lobiyal. "Fuzzy Hindi WordNet and word sense disambiguation using fuzzy graph connectivity measures." ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP) 15, no. 2: 1-31, 2015

[6]A. Jain, S. Vij, and O. Castillo. "Hindi query expansion based on semantic importance of Hindi WordNet relations and fuzzy graph connectivity measures." *Computación y Sistemas 23,* no. 4: 1337-1355, 2019.

[7]Jain, Minni, Grusha Bhalla, Amita Jain, and Swati Sharma. "Automatic keyword extraction for localized tweets using fuzzy graph connectivity measures." Multimedia Tools and Applications 81, no. 30 (2022): 42931-42956.

[8]A. Jain, and D. K. Lobiyal. "Unsupervised Hindi word sense disambiguation based on network agglomeration." In 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), pp. 195-200. IEEE, 2015.

[9] C. Cornelis , G. Deschrijver, and E. E. Kerre. "Implication in intuitionistic fuzzy and interval-valued fuzzy set theory: construction, classification, application." *International journal of approximate reasoning, 35(1), pp.55-95,* 2004.

[10] M. Akram, & W. A. Dudek. "Interval-valued fuzzy graphs." *Computers & Mathematics with Applications, 61(2), 289-299*, 2011.

[11] M. Akram, & W. A. Dudek. "Interval-valued fuzzy graphs." *Computers & Mathematics with Applications, 61(2), 289-299*, 2011.

[12]C. Diou, G. Katsikatsos, and A. Delopoulos. "Constructing Fuzzy Relations fromWordNet forWord Sense Disambiguation." *In 2006 First International Workshop on Semantic Media Adaptation and Personalization (SMAP'06),* pp. 135-140. IEEE, 2006

[13] A. Joshi , A. R. Balamurali, and P. Bhattacharyya. "A fall-back strategy for sentiment analysis in Hindi: a case study." *Proceedings of the 8th ICON*, 2010.

[14]P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. "Enriching word vectors with subword information." *Transactions of the association for computational linguistics 5*: 135-146, 2017.

[15] A. Jain, M. Jain "Detection and correction of non-word spelling errors in Hindi language". International Conference on Data Mining and Intelligent Computing (ICDMIC), IEEE,1-5, 2014.

[16] R. Saluja, D. Adiga, G. Ramakrishnan, P. Chaudhuri, M. Carman, "A Framework for Document Specific Error Detection and Corrections in Indic OCR). *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR),* IEEE 4:25-30, 2017.

[17] D. Gupta, and S. Bag. "Handwritten multilingual word segmentation using polygonal approximation of digital curves for Indian languages." Multimedia Tools and Applications 78: 19361-19386, 2019.

[18]S. Puri, and S. P. Singh. "An efficient hindi text classification model using svm." In Computing and Network Sustainability: Proceedings of IRSCNS 2018, pp. 227-237. Springer Singapore, 2019.

[19] S. Kanwar, M. Sachan, and G. Singh. "N-GRAMS SOLUTION FOR ERROR DETECTION AND CORRECTION IN HINDI LANGUAGE." International Journal of Advanced Research in Computer Science 8, no. 7, 2017.

[20] P. Etoori, M. Chinnakotla, and R. Mamidi. "Automatic spelling correction for resource-scarce languages using deep learning." In Proceedings of ACL 2018, Student Research Workshop, pp. 146-152, 2018.

[21] A.Jain, M. Jain, G. Jain, DK Tayal "UTTAM", An Efficient Spelling Correction System for Hindi Language Based on Supervised Learning. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* 18(1):8, 2018.

[22] R. Navigli, and M. Lapata. "Graph connectivity measures for unsupervised word sense disambiguation." In IJCAI, vol. 7, pp. 1683-1688, 2007.

[23] S. Vij,A. Jain,D. Tayal, O. Castillo. "Fuzzy Logic for Inculcating Significance of Semantic Relations in Word Sense Disambiguation Using a WordNet Graph". International Journal of Fuzzy Systems 20(2):444-459.2017.

[24] A. Miller, A. George, L.Claudia, T. Randee, and T. Ross. "A semantic concordance." In Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993.

[25] R.Izquierdo, P. Marten, and V.Piek . "Topic modeling and word sense disambiguation on the Ancora corpus." Procesamiento del Lenguaje Natural 55 : 15-22.2015.

[26] Yuan, Dayu, Julian Richardson, Ryan Doherty, Colin Evans, and Eric Altendorf. "Semi-supervised word sense disambiguation with neural models." arXiv preprint arXiv:1603.07012 (2016).

[27] Lesk, Michael. "Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone." In Proceedings of the 5th annual international conference on Systems documentation, pp. 24-26. 1986.

[28] G. Tsatsaronis, M. Vazirgiannis, and I. Androutsopoulos. "Word sense disambiguation with spreading activation networks generated from thesauri." *In IJCAI*, vol. 27, pp. 223-252, 2007.

[29] Navigli, Roberto, and Mirella Lapata. "An experimental study of graph connectivity for unsupervised word sense disambiguation." IEEE transactions on pattern analysis and machine intelligence 32, no. 4 (2009): 678-692.

[30] M. Kracker. "A fuzzy concept network model and its applications." *In [1992 Proceedings] IEEE International Conference on Fuzzy Systems*, pp. 761-768. IEEE, 1992.

[31] S. M. Chen, Y. J. Horng, and C. H. Lee. "Document retrieval using fuzzy-valued concept networks." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 31*, no. 1: 111-118, 2001.

[32] S. M. Chen, Y. J. Horng, and C. H. Lee. "Fuzzy information retrieval based on multi-relationship fuzzy concept networks." *Fuzzy Sets and Systems 140*, no. 1: 183-205, 2003.

[33] V. Kann and M. Rosell. "Free construction of a free Swedish dictionary of synonyms." *Proc. 15th Nordic Conf. on Comp. Ling.–NODALIDA (5),2005.*

[34] H.O. Gonçalo, and P. Gomes. "Automatic Discovery of Fuzzy Synsets from Dictionary Definitions." *22nd International Joint Conference on Artificial Intelligence, 1801–1806. http://ijcai.org/papers11/Papers/IJCAI11-302.pdf,2011.*

[35] Y. Alizadeh-Q, B. Minaei-Bidgoli, S. A. Hossayni, M.R. Akbarzadeh-T, D. R. Recupero, M. R. Rajati and A. Gangemi. "Interval Probabilistic Fuzzy WordNet." *arXiv preprint arXiv:2104.10660,* 2021

[36] Medhat, Walaa, A. Hassan, and H. Korashy. "Sentiment analysis algorithms and applications: A survey." *Ain Shams engineering journal 5, no. 4 (2014): 1093-1113.*

[37] A. Das, and S. Bandyopadhyay. "SentiWordNet for Indian languages." *In Proceedings of the eighth workshop on Asian language resouces, pp. 56-63, 2010.*

[38] A. Joshi, A. R. Balamurali, and P. Bhattacharyya. "A fall-back strategy for sentiment analysis in hindi: a case study." *Proceedings of the 8th ICON, 2010.*

[39] Bakliwal, Akshat, Piyush Arora, and Vasudeva Varma. "Hindi subjective lexicon: A lexical resource for hindi adjective polarity classification." *In Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), pp. 1189-1196. 2012.*

[40] Mittal, Namita, Basant Agarwal, Garvit Chouhan, Nitin Bania, and Prateek Pareek. "Sentiment analysis of hindi reviews based on negation and discourse relation." *In Proceedings of the 11th Workshop on Asian Language Resources, pp. 45-50. 2013.*

[41] A. Piyush, A. Bakliwal, and V. Varma. "Hindi subjective lexicon generation usingwordnet graph traversal." *International Journal of Computational Linguistics and Applications 3, 1 (2012), 25–39,* 2012.

[42] R. Sharma and P. Bhattacharyya. "A sentiment analyzer for Hindi using Hindi senti lexicon." *In Proceedings of the 11th International Conference on Natural Language Processing*, 2014

[43] V. Jha, N. Manjunath, P. D. Shenoy, and K. R. Venugopal. "Sentiment analysis in a resource scarce language: Hindi." *International Journal of Scientific & Engineering Research 7, 9 (2016), 968–990,* 2016.

[44] D. Mishra, M. Venugopalan, and D. Gupta. "Context specific Lexicon for Hindi reviews." *Procedia Computer Science 93 (2016), 554–563,* 2016

[45] M. Deepa and N. Nain. "Part-of-speech tagging of Hindi corpus using rule-based method." *In Proceedings of the International Conference on Recent Cognizance in Wireless Communication & Image Processing. Springer, 241–247,2016.*

[46] V. Jha, R. Savitha, P. D. Shenoy, K.R Venugopal, and A.K. Sangaiah. "A novel sentiment aware dictionary for multi-domain sentiment classification." *Computers & Electrical Engineering 69, (2018), 585–597,* 2018.

[47] F. Hussaini , S. Padmaja, and S. Sameen Fatima “Score-based sentiment analysis of book reviews in Hindi language.” *International Journal on Natural Language Computing 7, 5 (2018), 115–127*, 2018.

[48] Y. Sharma, V. Mangat, and M. Kaur. “A practical approach to sentiment analysis of Hindi tweets.” *In Proceedings of the 1st International Conference on Next Generation Computing Technologies. 677–680*, 2015.

[49] V. Jha , N. Manjunath, P. D. Shenoy, and K. R. Venugopal. “HSAS: Hindi subjectivity analysis system.” *In Proceedings of the 2015 Annual IEEE India Conference (INDICON). IEEE, 1–6*, 2015.

[50] Shapiro, A. Hale, M. Sudhof, and D. Wilson. 2020. “Measuring news sentiment. Journal of Econometrics (2020).” *DOI:https://doi.org/10.1016/j.jeconom.2020.07.053*

[51] K. Garg. “Sentiment analysis of Indian PM’s ‘Mann Ki Baat’.” *International Journal of Information Technology 12, 1 (2019), 37–48*, 2019.

[52] R. Sharma, S. Nigam, and R. Jain.“Polarity detection movie reviews in Hindi language.” *International Journal on Computational Sciences & Applications (IJCSA) 4, 4 (2014), 49–57*, 2014.

[53] V. Gupta, N. Jain, S. Shubham, A. Madan, A. Chaudhary, and Q. Xin. "Toward integrated cnn-based sentiment analysis of tweets for scarce-resource language— Hindi." *Transactions on Asian and Low-Resource Language Information Processing 20*, no. 5: 1-23, 2021.

[54] S. Ghosal, and A. Jain. "HateCircle and Unsupervised Hate Speech Detection Incorporating Emotion and Contextual Semantics." *ACM Transactions on Asian and Low-Resource Language Information Processing 22*, no. 4: 1-28, 2023.

[55] S.K. Biswas, M. Bordoloi, and J. Shreya. “A graph-based keyword extraction model using collective node weight.” *Expert Systems with Applications*, 2018. https://doi.org/10.1016/j.eswa.2017.12.025

[56] Y. Matsuo, T. Sakaki, K. Uchiyama, and M. Ishizuka *“*Graph-based word clustering using a web search engine.*” COLING/ACL 2006 - EMNLP 2006: 2006 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference,* 2006. https://doi.org/10.3115/1610075.1610150

[57] R. Barzilay, and M. Elhadad. *“*Using lexical chains for text summarization.Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization.*”,* 1997. https://doi.org/10.3115/1034678.1034760

[58] H.G. Silber, and K.F. McCoy. *“*Efficient text summarization using lexical chains*”. International Conference on Intelligent User Interfaces, Proceedings IUI,* 2000. https://doi.org/10.3115/1118253.1118294

[59] T. Simone, and M. Moens. “Summarizing Scientific Articles - Experiments with Relevance and Rhetorical Status.” *Computational Linguistics*, 2002.

[60] M.R. Amini, and P. Gallinari. “The use of unlabelled data to improve supervised learning for text summarization.” *SIGIR Forum (ACM Special Interest Group on Information Retrieval).”,* 2002. https://doi.org/10.1145/564396.564397

[61] N. Begum, M.A. Fattah, and F. Ren. *“*Automatic text summarization using support vector machine.” *International Journal of Innovative Computing, Information and Control.”* ,2009.

[62] C. Fang, D. Mu, Z. Deng, and Z. Wu. “Word-sentence co-ranking for automatic extractive text summarization.” *Expert Systems with Applications,2017.* https://doi.org/10.1016/j.eswa.2016.12.021

[63] D. Wang, T. Li, S. Zhu, and C. Ding. “Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization.” *ACM SIGIR 2008 - 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Proceeding, 2008.* https://doi.org/10.1145/1390334.1390387

[64] H. Christian, M.P. Agus, and D. Suhartono. “Single Document Automatic Text Summarization using Term Frequency-Inverse Document Frequency (TF-IDF)”. *ComTech: Computer, Mathematics and Engineering Applications, 2016.* https://doi.org/10.21512/comtech.v7i4.3746

[65] J.M. Conroy, J.D. Schlesinger, J. Goldstein, and D.P. O’Leary. “Left-Brain/Right-Brain Multi-Document Summarization.” *Proceedings of DUC 2004, 4th Document Understanding Conference,* 2004.

[66] A. Aries, D.E. Zegour, and K.W. Hidouci. “All Summarizer system at multiLing 2015: Multilingual single and multi-document summarization.” *SIGDIAL 2015 - 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, Proceedings of the Conference*, 2015. https://doi.org/10.18653/v1/w15-4634

[67] S. Sripada, V. G. Kasturi and G. K. Parai. “Multi-document extraction-based Summarization *.” *A Course Project*, 2010.

[68] L. Page, S. Brin, R. Motwani, and T. Winograd. “The PageRank Citation Ranking: Bringing Order to the Web. World Wide Web Internet And Web Information Systems.”,1998. https://doi.org/10.1.1.31.1768

[69] J.M. Kleinberg. “Authoritative sources in a hyperlinked environment. In The Structure and Dynamics of Networks.”*, 2011.* https://doi.org/10.1515/9781400841356.514

[70] T. Hirao, Y. Yoshida, M. Nishino, N. Yasuda, and M. Nagata. “*Single-document summarization as a tree knapsack problem.”* EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference. (2013).

[71] T. Berg-Kirkpatrick, D. Gillick and D. Klein. “Jointly learning to extract and compress.” *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies,*2011.

[72] D. Parveen, H. M. Ramsl, and M. Strube. “Topical coherence for graph-based extractive summarization.” *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing*, 2015, https://doi.org/10.18653/v1/d15-1226

[73] J. Steinberger, and K. Jezek. “Using latent semantic analysis in text summarization and summary evaluation” *Proceedings of the International Conference on Information System Implementation and Modelling (ISIM’04),* 2004.

[74] A. Haghighi, and L. Vanderwende. “Exploring content models for multi-document summarization.” *NAACL HLT 2009 - Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, 2009. https://doi.org/10.3115/1620754.1620807

[75] X. Wan, and J. Xiao. “Exploiting neighborhood knowledge for summarization and keyphrase extraction.” *ACM Transactions on Information Systems*, 2010. https://doi.org/10.1145/1740592.1740596

[76] D. Shen, J.T. Sun, H. Li, Q. Yang, and Z. Chen. *"*Document summarization using conditional random fields*." IJCAI International Joint Conference on Artificial Intelligence*", 2007.

[77] D. Galanis, G. Lampouras, and I. Androutsopoulos *"*Extractive multi-document summarization with integer linear programming and support vector regression." *24th International Conference on Computational Linguistics - Proceedings of COLING 2012: Technical Papers*, 2012.

[78] J. Carbonell, J., and J. Goldstein "Use of MMR, diversity-based reranking for reordering documents and producing summaries." *SIGIR Forum (ACM Special Interest Group on Information Retrieval),1998.* https://doi.org/10.1145/3130348.3130369

[79] A.S. Parihar, A. Jain, and A. Gupta. "Citation-Based Scientific Paper Summarization Using Game Theory." *In 2020 5th International Conference on Communication and Electronics Systems (ICCES) (pp. 1157-1161). IEEE,* 2020.

[80] J. Cheng and M. Lapata. "Neural summarization by extracting sentences and words." *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, 2016, https://doi.org/10.18653/v1/p16-1046.

[81] R. Ferreira, L. De Souza Cabral, R. D. Lins, G. Pereira E Silva, F. Freitas, G. D. C. Cavalcanti, R. Lima, S. J. Simske and L. Favaro. "Assessing sentence scoring techniques for extractive text summarization." *In Expert Systems with Applications,* 2013, https://doi.org/10.1016/j.eswa.2013.04.023

[82] B. Mutlu, E. A. Sezer and M. A. Akcayol. "Multi-document extractive text summarization: A comparative assessment on features." *Knowledge-Based Systems*, 2019, https://doi.org/10.1016/j.knosys.2019.07.019.

[83] S. Gupta and S. K. Gupta. "Abstractive summarization: An overview of the state of the art." *In Expert Systems with Applications*, 2019, https://doi.org/10.1016/j.eswa.2018.12.011.

[84] X. Wan. "Towards a unified approach to simultaneous single-document and multi-document summarizations." *Coling 2010 - 23rd International Conference on Computational Linguistics, Proceedings of the Conference*, 2010.

[85] P. Verma and H. Om. "A variable dimension optimization approach for text summarization." *Advances in Intelligent Systems and Computing*, 2019, https://doi.org/10.1007/978-981-13-0761-4_66.

[86] M. Jain, S. Bansal, and Y. Gupta. "Experimental Comparison and Scientometric Inspection of Research for Word Embeddings." In Proceedings of 3rd International Conference on Computing Informatics and Networks: ICCIN 2020, pp. 3-11. Springer Singapore, 2021.

[87] B. Han, P. Cook, and T. Baldwin. "Automatically constructing a normalisation dictionary for microblogs." In Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning, pp. 421-432, 2012.

[88] Etoori P, Chinnakotla M, Mamidi R (2018) Automatic Spelling Correction for Resource-Scarce Languages using Deep Learning.", In *Proceedings of ACL 2018, Student Research Workshop*, pp. 146-152.

[89] Pal, Alok Ranjan, and Diganta Saha. "Word sense disambiguation: A survey." arXiv preprint arXiv:1508.01346 (2015).

[90] L. Becchetti, C. Carlos. "The distribution of PageRank follows a power-law only for particular values of the damping factor." In Proceedings of the 15th international conference on World Wide Web, pp. 941-942. 2006.

[91] B. Snyder, P. Martha. "The English all-words task." In Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, pp. 41-43. 2004.

[92] K. Abdalgader, and A. A. Shibli. "Context expansion approach for graph-based word sense disambiguation." *Expert Systems with Applications 168*: *114313,* 2021.

[93] M. AlMousa, R. Benlamri, and R. Khoury. "A novel word sense disambiguation approach using WordNet knowledge graph." *Computer Speech & Language 74*: 101337, 2022.

[94] G. A. Miller. "WordNet: a lexical database for English." *Communications of the ACM 38,* no. 11: 39-41, 1995.

[95] D. Narayan, D. Chakrabarti, P. Pande, and P. Bhattacharyya. "An experience in building the indo wordnet-a wordnet for hindi." *In First international conference on global WordNet*, Mysore, India, vol. 24. 2002.

[96] A. Das, and S. Bandyopadhyay. "SentiWordNet for Indian languages." *In Proceedings of the eighth workshop on Asian language resouces*, pp. 56-63. 2010.

[97] J. Pennington, R. Socher, and C. D. Manning. "Glove: Global vectors for word representation." *In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP),* pp. 1532-1543, 2014.

[98] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv*:1810.04805, 2018.

[99] Cambria, Erik, and A. Hussain. "SenticNet." In Sentic Computing*, pp. 23-71. Springer, Cham, 2015*.

[100] M. Thelwall. "The Heart and soul of the web? Sentiment strength detection in the social web with SentiStrength." *In Cyberemotions, pp. 119-134. Springer, Cham, 2017.*
[101] V. Gaikwad, and Y. Haribhakta. "Adaptive glove and fasttext model for hindi word embeddings." *In Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, pp. 175-179, 2020.

[102] A. Pathak, S. Kumar, P. P. Roy, and B.G. Kim. "Aspect-based sentiment analysis in Hindi language by ensembling pre-trained mBERT models." *Electronics 10*, no. 21: 2641, 2021.

[103] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam et al. "Muril: Multilingual representations for indian languages." *arXiv preprint arXiv:*2103.10730, 2021

[104] S. M. Mohammad, and P. D. Turney. "Nrc emotion lexicon." *National Research Council,* Canada 2: 234, 2013.

[105] R. Bose, R. K. Dey, S. Roy, and D. Sarddar. "Analyzing political sentiment using Twitter data." *In Information and communication technology for intelligent systems*, pp. 427-436. Springer, Singapore, 2019

[106] A. R. Balamurali, A. Joshi and P. Bhattacharyya. “Cross-lingual sentiment analysis for Indian languages using linked wordnets.” *Proceedings of Coling*, 73–82, 2012.

[107] S.A. Taher, K.A. Akhter and K.A. Hasan. “N-gram based sentiment mining for bangla text using support vector machine.” *In 2018 international conference on Bangla speech and language processing (ICBSLP)* (pp. 1-5). IEEE, September 2018.

[108] E. Sivasankar, K. Krishnakumari, and P. Balasubramanian. “An enhanced sentiment dictionary for domain adaptation with multi-domain dataset in Tamil language (ESD-DA).” *Soft Computing*, 25(5), pp.3697-3711, 2021.

[109] M. A. Nowak, N. L. Komarova and P. Niyogi. “Evolution of universal grammar.” *Science*, 2001, https://doi.org/10.1126/science.291.5501.114.

[110] R. Tripodi and M. Pelillo. “A game-theoretic approach to word sense disambiguation.” *Computational Linguistics*, 2017, https://doi.org/10.1162/COLI_a_00274.

[111] M. Jain, A. Suvarna and A. Jain. “An evolutionary game theory based approach for query expansion.” *Multimedia Tools and Applications*, pp.1-25, 2022.

[112] A. Choudhry, I. Khatri, M. Jain and D.K. Vishwakarma. *“*An Emotion-Aware Multitask Approach to Fake News and Rumor Detection Using Transfer Learning*.” IEEE Transactions on Computational Social Systems*, 2002.

[113] M. Jain, A. Jaswani, A. Mehra, and L. Mudgal. “EDGly: detection of influential nodes using game theory*.” Multimedia Tools and Applications, pp.1-23.Egger, R. (2022) Text Representations and Word Embeddings: Vectorizing Textual Data." In Applied Data Science in Tourism: Interdisciplinary Approaches, Methodologies, and Applications, pp. 335-361. Cham: Springer International Publishing*, 2022.

[114] R. Mihalcea, and P. Tarau. “TextRank: Bringing order into texts.” *Proceedings of EMNLP, 2004.* https://doi.org/10.3115/1219044.1219064

[115] A. Nenkova, and K. McKeown. “Automatic summarization. In Foundations and Trends in Information Retrieval.”, 2011. https://doi.org/10.1561/1500000015

[116] G. Erkan and D. R. Radev. “LexRank: Graph-based lexical centrality as salience in text summarization.” *Journal of Artificial Intelligence Research*, 2004, https://doi.org/10.1613/jair.1523.

[117] J. Steinberger and K. Ježek. “Text summarization and singular value decomposition.” *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* 2004, https://doi.org/10.1007/978-3-540-30198-1_25.

[118] K. Woodsend, and M. Lapata. “Learning to simplify sentences with quasi-synchronous grammar and integer programming*.” EMNLP 2011 - Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2011.

[119] A. Joshi, E. Fidalgo, E. Alegre, and L. Fernández-Robles*. “*SummCoder: An unsupervised framework for extractive text summarization based on deep auto-encoders*.” Expert Systems with Applications, 129, pp.200-215,* 2019.

[120] L. Sterckx, T. Demeester, J. Deleu, and C. Develder. "Topical word importance for fast keyphrase extraction." *In Proceedings of the 24th International Conference on World Wide Web, page 121–122. Association for Computing Machinery*, 2015.

[121] A. Bougouin, F. Boudin, and B. Daille. "Topicrank: Graph-based topic ranking for keyphrase ´ extraction." *In Proceedings of the Sixth International Joint Conference on Natural Language Processing, pages 543–551. Asian Federation of Natural Language Processing*, 2013.

[122] A. Ahmad, and T. Ahmad. "A game theory approach for multi-document summarization." *Arabian Journal for Science and Engineering*, *44(4), pp.3655-3667*, 2019.

[123] A. Gupta, M. Kaur, S. Mittal, and S. Garg. "PE-MSC: partial entailment-based minimum set cover for text summarization." *Knowledge and Information Systems, 63(5), pp.1045-1068*, 2021.