Word Sense Disambiguation for Punjabi Language

A Thesis

Submitted in the fulfilment of the requirements for the award of the degree of

Doctor of Philosophy

Submitted by

Varinder Pal Singh

(Registration No. 951203003)

Under the Supervision of

Dr. Parteek Kumar

Professor

Computer Science and Engineering Department,

Thapar Institute of Engineering and Technology, Patiala

January 2022

Certificate

I hereby certify that the work which is being submitted in this thesis entitled "*Word sense disambiguation for Punjabi language*", in the fulfilment of the requirements for the award of the degree of DOCTOR OF PHILOSOPHY submitted in Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala is an authentic record of my own work carried out under the supervision of Dr. Parteek Kumar and refer work of other researchers which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of degree in any other University.

Varinder Pal Singh

Reg. No 951203003

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge and belief.

Parteek Kumar

Parteek Kumar

Professor, Computer Science and Engineering Department,

Thapar Institute of Engineering and Technology, Patiala-147004 Punjab

Supervisor

Acknowledgement

First of all, I express my gratitude to the Almighty, who blessed me with the zeal and enthusiasm to complete this work successfully.

I would like to thank my supervisor Dr. Parteek Kumar for his valuable guidance. He has always been there to guide me towards the right direction whenever I got stuck with the ideas. I want to thank him for his valuable evenings and week-ends that he spent with me in finalizing the writings of this thesis work. I also want to thank Mrs. Kumar to provide a conductive environment in her home during discussions on this proposed work.

I express my gratitude to the Doctoral Committee for monitoring the progress and providing valuable suggestions for improvement of my research work. I feel very proud and thankful to the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala for providing all the resources for a good research work. I am also very thankful to Dr. Maninder Singh (Head CSE department, TIET, Patiala) and all my colleagues for their constant support and cooperation during my research work.

I take this opportunity to thank my family members, who are always there to support me morally and emotionally.

(Varinder pal Singh)

Abstract

The recent eruption of data in different natural languages on the internet has necessitated the development of Natural Language Processing (NLP) tasks. The major impediments in the development and implementation of NLP are the scarcity of the standard datasets, knowledge resources, language tools and ambiguity resolution. Word Sense Disambiguation (WSD) is a critical and essential task for machine translation, information retrieval, question answering and sentiment analysis, *etc*. NLP tasks.

The objective of WSD is to automatically select the appropriate sense of an ambiguous word based on the context of the word. The WSD process identifies the different senses for every word relevant to the text or discourse under consideration from the sense inventories such as dictionaries, thesaurus, ontologies, and WordNet. Then it involves a mean to assign the appropriate sense to each occurrence of a word in context. Thus WSD needs the representation of common sense and encyclopaedic knowledge to resolve the sense of ambiguous words. Recognizing the proper sense of a word in context by a computer is defined as an AI-complete complexity problem. There are two different types of WSD, namely targeted WSD and all-words WSD. The targeted WSD resolves the ambiguity of an ambiguous target word, usually occurring one per sentence. The all-words WSD disambiguates all open-class words (noun, adverb, verb, adjectives) in a text. In this research work, targeted WSD was implemented.

India is a multilingual country having 22 national languages. Interlanguage processing tasks like machine translation, question answering, sentiment analysis, cross-lingual search, *etc*., are highly applicable problems in India. The Punjabi language is the official language of the Indian state of Punjab, and it is the world's 10^{th} most widely spoken language. The Punjabi diasporas are present globally, and there is a need for the Punjabi NLP tasks to connect them to the Punjabi language successfully. It motivated us to explore the field of WSD for the Punjabi language. WSD has been successfully designed and developed for the English language. However, there are many differences in the language structure of English and Punjabi, which arise different challenges while performing sense disambiguation on the Punjabi text dataset.

In this research work, the systematic review has explicitly portrayed WSD in Indian languages. A review methodology has followed with the help of the framed research questions. The renowned electronic databases and the topmost conferences were explored to include the relevant studies of WSD for Indian languages. The existing status of the WSD for Indian

languages has categorised as per the different families of the Indian languages. The evolution of WSD for Indian languages and their publication time is reported. This research has reviewed and analysed the WSD for Indian languages based on the techniques, knowledge resources and evaluation methods. The review of the standard Senseval/Semeval evaluation workshops for WSD field has been presented. The findings of this research work, such as the available raw corpora, sense tagged corpus, dictionaries, WordNet and pre-processing linguistic tools of different Indian languages, will help the researchers. The comprehensive survey presented in this research work would assist the researchers in choosing the most suitable WSD in the specific domain and the pertinent future directions. The availability of the Punjabi WordNet has motivated us to implement the Punjabi WSD.

In this thesis, the supervised Punjabi WSD has designed and developed. The supervised approach for WSD is the most explored, but it needs a sense tagged dataset of a language. A sense tagged dataset for 150 most ambiguous Punjabi noun words has been prepared, to accomplish the supervised Punjabi WSD. The availability of the Punjabi WordNet, a lexical knowledge resource, has helped us to select and develop the Punjabi sense tagged dataset for the 150 most ambiguous Punjabi nouns. The senses of the ambiguous words returned by the Punjabi WordNet are overlapping, also called fine senses. The fine senses have been merged by applying rules to produce more discerning or coarse senses of the ambiguous Punjabi word. For the coarse senses of the 150 ambiguous words, data has been collected from the online and offline Punjabi language resources. The different senses of the individual Punjabi ambiguous words have been tagged to the collected dataset by three native speakers of the Punjabi language. The sense tagged dataset of the individual ambiguous word has been validated with the help of the kappa inter-annotator agreement.

The modular architecture of the WSD system for the Punjabi language has been laid out to implement the explicit Punjabi WSD. The objective of this Punjabi WSD system is to ascertain the sense of the ambiguous Punjabi word independent of any NLP task. The individual phases/modules of the Punjabi WSD architecture are the data collection, pre-process, parsing, feature extraction and classification task. The pre-processing of the Punjabi sense tagged dataset involves the removal of unwanted noise such as nonstandard fonts, punctuations and multiple spellings. The Punjabi abbreviation completion dictionary, a standard Punjabi spelling dictionary and Punjabi stopwords dictionary are the outcomes of the pre-processing phase of the architecture of the Punjabi WSD. The open-source Punjabi shallow parser has been applied to the sense tagged Punjabi dataset to extract the root words, PoS tags and headwords.

This research has explored the topical, local, syntactic and semantic text features for the individual Punjabi ambiguous word. The sound evidence (the text features) has selected from the context of the target Punjabi ambiguous words. The categorization of the features is based on the diverse combinations of words and their respective grammatical relations. The topical (bag of words and co-occurring words) and the local (collocations and n-grams) features have derived from the sense tagged dataset of the individual Punjabi ambiguous word. The syntactic (part of speech tags) features have selected from the PoS tagged dataset. The open-source Punjabi shallow parser assigns the PoS tags to the Punjabi sense tagged dataset. The semantic count-based features prepare the word × document co-occurrence matrice for each ambiguous word from its sense tagged dataset. The predictive word embedding features apply the neural network models (Word2Vec, Global Vector and FastText) on the unlabelled Punjabi language corpus to generate the dense and fixed dimension vectors. The Punjabi language word embedding features have been checked against the similarity and analogy completion tasks.

Finally, the Punjabi WSD task, a two-step classification has been implemented. In the first step, the Machine Learning (ML) and deep learning classifiers have been trained using topical, local and syntactic features. The nine ML classifiers Decision List, Decision Tree, Random Forest, Naïve Bayes, Support Vector Machine based on sequential minimal optimisation, k-Nearest Neighbour, Adaboost, Stacking, StackingC, have been trained individually on the topical, local, syntactic and semantic features. The topical, local and syntactic features are the collection of strings values. These string features have been transformed into numeric vectors before the training of the ML classifiers. Whereas the two semantic features (count-based and prediction-based co-occurrence matrices) of the ambiguous word have been processed using concatenation, average, fractional and exponential vector combination approaches to train the ML classifiers. The deep learning classifiers, namely multi-layer perceptron and long short term memory, have been directly trained on the prediction-based word embedding features. In the second step, the trained machine learning and deep learning classifiers predict the most practical sense of the ambiguous word. The Punjabi WSD system has been integrated with a web interface and is available online at www.punjabiwsd.in.

The evaluation of the predictive performance of the explicit Punjabi WSD system for the individual ambiguous word has been measured using the accuracy metric. The estimated accuracies of the individual word using supervised classifiers on lexical and sematic feature sets are exceeding the baseline accuracy (the most frequent sense). The accuracy of the classifiers is poor when the syntactic features have used. It has been observed that syntactic

tagging is dependent on the Part of Speech (PoS) tagger. The analysis of the PoS tags of the Punjabi text shows the high error rate in the PoS tagging. The results of this experiment show that the words having more senses are difficult to disambiguate. The words having an enormous sense labelled dataset produce better predictions for the unknown instances. The words having a skewed dataset are prone to the MFS assignment anomaly. The ensemble classifiers produce better performance for the skewed dataset of the Punjabi noun words. This supervised Punjabi WSD has yielded a significantly high accuracy than the existing knowledge base classifiers.

The performance of the Punjabi WSD system is encouraging, and it has the potential to provide impetus to the development of different NLP tasks of the Punjabi language. The Punjabi WSD can be used for the benefit of society.

List of Figures

Figure 1.1	Explaining the WSD task	2
Figure 1.2	Implicit and explicit usage of WSD	3
Figure 1.3	The process of word sense disambiguation	4
Figure 1.4	Different types of the knowledge resources	6
Figure 1.5	Classification approaches used for WSD	9
Figure 1.6	Types of knowledge base approaches of WSD	11
Figure 1.7	Types of unsupervised approaches of WSD	13
Figure 2.1	Review procedure followed	50
Figure 2.2	Indian language families (Chand, 2016)	51
Figure 2.3	Evolution of the WSD for Indian languages	52
Figure 2.4	Trends in the publications of the WSD for Indian languages	53
Figure 2.5	Status of the WSD research in different families of Indian languages	72
Figure 2.6	Status of the WSD research in different Indian languages	73
Figure 2.7	Use of WordNet in the WSD approaches for Indian languages	78
Figure 2.8	WSD approaches applied for Indian languages	79
Figure 2.9	Supervised WSD approaches applied for Indian languages	80
Figure 2.10	Types of the WSD system for Indian languages	81
Figure 3.1	Sense tagged dataset preparation process	90
Figure 3.2	Relations of synset ਘਰ (TT: ghar) selected from the Punjabi WordNet (Kaur, et al., 2010)	96
Figure 3.3	Distribution of PoS word classes in the Punjabi WordNet	97
Figure 3.4	Fine and coarse senses of 150 ambiguous words	101
Figure 3.5	Kappa statistics of 150 ambiguous Punjabi noun words	107
Figure 4.1	The Architecture of the Punjabi WSD	116
Figure 4.2	The basic decision tree structure	128
Figure 4.3	Hyperplane example for the SVM (Navigli, 2009)	131
Figure 4.4	Multilayer feed forward NN	135
Figure 4.5	The basic gate of the LSTM	137
Figure 5.1	Different features selected for the Punjabi WSD	144
Figure 5.2	Process to generate the Punjabi word vectors	157
Figure 5.3	CBOW model with one-word context	159
Figure 5.4	CBOW model with multi-word context	160
Figure 5.5	Skip Gram Word2Vec model	161

Figure 6.1	Supervised Learning	174
Figure 6.2	The setup of the LSTM model	185
Figure 6.3	Flowchart of predicting sense of the ambiguous words using trained classifiers	188
Figure 6.4	JSON file structure used to store the Punjabi ambiguous words and their coarse senses	190
Figure 6.5	Sense predicted by the Punjabi WSD	190
Figure 6.6	Sense predicted by the Punjabi WSD of multiple words	191
Figure 6.7	Feature specific Punjabi WSD	192
Figure 6.8	Analysis of the Word embedding features	193
Figure 7.1	MFS accuracy of 150 ambiguous Punjabi noun words	199
Figure 7.2	Average accuracies of the ML classifiers using bigram features	202
Figure 7.3	Average accuracies of the ML classifiers using four combination methods for CBoW and SG word vectors	205
Figure 7.4	Distribution of accuraices yielded by ML classifiers using CBoW word vector with FD combination for Punjabi datase	206
Figure 7.5	Distribution of accuraices yielded by ML classifiers using SG word vector with FD combination for Punjabi dataset	207

List of Tables

Table 1.1	Different types of the features based on the context representation	9
Table 1.2	Punjabi Letters in the Gurmukhi Script	27
Table 1.3	Free word order examples	28
Table 1.4	Examples of simple, complex and compound Punjabi sentences	28
Table 1.5	Punjabi noun word with its case, number and gender	29
Table 1.6	Example of ablative case of Punjabi preposition	31
Table 1.7	Examples of English words used in Punjabi sentences	32
Table 1.8	Null subject Punjabi sentences	33
Table 1.9	Gender overlap examples of Punjabi sentences	33
Table 1.10	Reflexive pronoun examples	34
Table 1.11	Newspaper headlines with same noun and verb examples	35
Table 2.1	Research questions for systematic literature review	47
Table 2.2	Search strings used to find the studies from e-resources	48
Table 2.3	Publication type of research studies	49
Table 2.4	Summarized details of the WSD for Indian languages	63
Table 2.5	Raw corpus available for Indian languages	76
Table 2.6	Sense annotated corpus for Indian languages	77
Table 2.7	Tools for Indian languages	79
Table 2.8	Supervised WSD approaches applied for Indian languages	81
Table 2.9	Senseval/Semeval workshop details	83
Table 3.1	Different meaning of the Punjabi ambiguous word ਉੱਤਰ (TT: uttar)	90
Table 3.2	Gloss and examples of different senses ids of word ਜਰ (TT: jar) selected from the Punjabi WordNet	92
Table 3.3	The details of words classes having different senses in the Punjabi WordNet	97
Table 3.4	Word senses of ਅੱਕ (TT: akk) defined in the Punjabi WordNet	98
Table 3.5	Applying sense reduction rules on the fine senses of word ਝਾੜ (TT: jhāṛ)	100
Table 3.6	Distribution of 150 ambiguous words based on their senses	102
Table 3.7	Selection of sense instances based on annotator agreement	104
Table 3.8	Details of the sense tagged files prepared for word ਜੱਗ (TT: jagg)	105
Table 3.9	Confusion matrix for the word ਫਲ (TT: phal) between two annotators	106
Table 3.10	Details of sense tagged dataset collected for 150 ambiguous words	108
Table 3.11	Details of training dataset collected for 150 ambiguous words	112
Table 4.1	Punjabi examples of abbreviations and their full forms	118

Table 4.2	Common misspelled Punjabi words	119
Table 5.1	Context processing of a Punjabi language sentence	142
Table 5.2	Stop words of the Punjabi language	143
Table 5.3	Number of instances corresponding to different senses of word ਸੂਆ (TT: sūā)	145
Table 5.4	Selected BoW features for different senses of word ਸੂਆ (TT: sūā)	146
Table 5.5	Collocation features of word ਸੂਆ (TT: sūā)	149
Table 5.6	Bigram features of word ਸੂਆ (TT: sūā)	150
Table 5.7	PoS tags for all senses of word ਸੂਆ (TT: sūā)	152
Table 5.8	PoS features of word ਸੂਆ (TT: sūā)	153
Table 5.9	word × document co-occurrence matrix of the ambiguous word ਸੂਆ (TT: sūā)	155
Table 5.10	List of the Punjabi unlabelled corpus	158
Table 5.11	Hyper parameters for Word2Vec model	166
Table 5.12	Most similar words extracted from Word2Vec (SG) word embedding vectors	167
Table 5.13	Most similar words extracted from GloVe word embedding vectors	168
Table 5.14	Most similar words extracted form Fatext word embedding vectors	169
Table 5.15	Comparison of Analogy completion task	170
Table 6.1	Hardware configuration of the computer machine	182
Table 6.2	Libraries explored for the training of the deep learning classifier	183
Table 6.3	Parameter of the input layer	184
Table 6.4	Parameters used by deep learning models	187
Table 7.1	Average accuracies of the nine ML classifiers using the BoW and Co-occurrence features	200
Table 7.2	Individual sense prediction by the SVM and StkC classifiers using BoW features of the Punjabi words	201
Table 7.3	Average accuracies of the nine ML classifiers using collocation and bigram feature sets	201
Table 7.4	Average accuracies of nine classifiers using PoS syntactic features	203
Table 7.5	Average accuracies of the ML classifiers using the word × document semantic feature	204
Table 7.6	Average accuracies of the ML classifiers using Word2Vec word vectors and four combination techniques	204
Table 7.7	Average accuracies produced by the ML classifiers using GloVe word vectors	207
Table 7.8	Average accuracies produced by the ML classifiers using FT word vectors	208
Table 7.9	Average accuracies of deep learning classifiers using word index vector	209

Table 7.10	Average accuracies of deep learning classifiers using word embedding vectors	209
Table 7.11	Predictive performance of the MLP and LSTM on the test dataset	210
Table 7.12	Average accuracies of deep learning classifiers using word embedding vectors	210
Table 7.13	List of the Punjabi WSD systems	212

List of Algorithms

Algorithm 5.1	Bag of words feature generation	146
Algorithm 5.2	Co-occurring words generation	147
Algorithm 5.3	Collocations selection	149
Algorithm 5.4	Bigram Selection	151
Algorithm 5.5	Selection of PoS combinations	153
Algorithm 6.1	Transformations of strings to binary vectors of BoW and co-occurrence string vectors	175
Algorithm 6.2	Transformations of strings to binary vectors using Collocation word vectors	177
Algorithm 6.3	Transformations of strings to binary vectors using Bigram word vectors	178
Algorithm 6.4	Transformation of strings to binary vectors using PoS vectors	178

Table of Contents

CERTIFICATE ..... ERROR! BOOKMARK NOT DEFINED.

ACKNOWLEDGEMENT I

ABSTRACT III

LIST OF FIGURES VII

LIST OF TABLES IX

LIST OF ALGORITHMS XII

CHAPTER 1 : INTRODUCTION TO WORD SENSE DISAMBIGUATION 1

1.1 OVERVIEW OF THE WSD 1
1.1.1 WSD a Classification Task *2*
1.1.2 Different Types of the WSD *2*
1.2 WSD PROCESS 4
1.2.1 Knowledge Resources *5*
1.2.2 Sense Representation *8*
1.2.3 Context Representation *9*
1.2.4 WSD Classification Techniques *10*
1.3 APPLICATIONS OF WSD 21
1.4 CHALLENGES IN DEVELOPMENT OF WSD SYSTEM 24
1.5 NEED OF WSD SYSTEM FOR THE PUNJABI LANGUAGE 26
1.5.1 Characteristics of the Punjabi Language *26*
1.5.2 Diversion of Punjabi over the English Language *32*
1.6 GAP ANALYSIS 35
1.7 RESEARCH OBJECTIVES 36
1.8 RESEARCH METHODOLOGY 37
1.9 RESEARCH CONTRIBUTIONS 39
1.10 THESIS ORGANISATION 40

CHAPTER 2 : LITERATURE SURVEY 45

2.1 EVOLUTION OF THE WSD 45
2.2 REVIEW METHODOLOGY FOLLOWED 46
2.2.1 Search Resources and Criteria *47*
2.2.2 Inclusion Exclusion Process *48*
2.3 INDIAN LANGUAGES AND THEIR FAMILIES 50
2.3.1 Evolution of WSD for Indian languages *52*
2.4 EXTRACTION OUTCOMES 52
2.4.1 Publication Timeline of the WSD for Indian Languages *52*
2.4.2 WSD for Indian Languages *53*
2.4.3 Indian Languages Explored for the WSD *72*
2.4.4 Resources for Implementation of the WSD *73*
2.4.5 Context Processing Resources *78*
2.4.6 Approaches for the WSD *79*
2.4.7 Evaluation Metrics for the WSD *82*
2.5 FINDINGS OF THE SYSTEMATIC REVIEW 84

CHAPTER 3 : DATA COLLECTION AND CORPUS CREATION 89

3.1 PUNJABI SENSE TAGGED CORPORA CREATION PROCESS 89
3.2 SELECTION OF THE PUNJABI AMBIGUOUS WORDS 90

3.2.1 Punjabi WordNet ... *91*
3.2.2 Analysis of the Punjabi WordNet ... *96*
3.3 MERGE FINE SENSE TO COARSE SENSE ... 98
3.4 SENSE TAGGING PROCESS ... 102
3.4.1 Data Collection for the Ambiguous Words ... *102*
3.4.2 Inter-Annotator Agreement of Sense Tagged Dataset ... *106*
3.4.3 Sense Tagged Punjabi Language Dataset ... *108*

CHAPTER 4 : THE ARCHITECTURE OF PUNJABI WSD ... 115

4.1 THE ARCHITECTURE OF THE PUNJABI WSD ... 115
4.1.1 Data collection ... *116*
4.1.2 Pre-processing ... *117*
4.1.3 Parsing ... *119*
4.1.4 Feature Extraction ... *124*
4.1.5 Classification ... *125*

CHAPTER 5 : FEATURES SELECTION ... 141

5.1 FEATURE SELECTION STEPS ... 141
5.1.1 Context Representation ... *141*
5.1.2 Stop Word List ... *143*
5.2 TYPES OF FEATURES ... 143
5.2.1 Most Frequent Sense ... *144*
5.2.2 Topical Features ... *145*
5.2.3 Local Features ... *148*
5.2.4 Syntactic Feature ... *151*
5.2.5 Semantic Features ... *154*

CHAPTER 6 : IMPLEMENTATION OF THE PUNJABI WSD ... 173

6.1 IMPLEMENTATION OF CLASSIFICATION TASK FOR PUNJABI WSD SYSTEM ... 173
6.1.1 String to Vector Transformations ... *174*
6.1.2 Training of Classifiers ... *181*
6.1.3 Predicting the Sense of the Ambiguous Word ... *187*
6.2 HOSTING OF THE PUNJABI WSD ON WEB ... 189
6.2.1 Data Structures Used in the Implementation of the Punjabi WSD ... *189*
6.2.2 The Punjabi WSD Interface ... *191*

CHAPTER 7 : RESULTS AND DISCUSSIONS ... 197

7.1 EVALUATION OF WSD ... 197
7.1.1 Evaluation Metric ... *197*
7.1.2 Baseline Most Frequent Sense (MFS) ... *198*
7.2 EXPERIMENTED RESULTS OF THE ML CLASSIFIERS ... 199
7.2.1 Analysis of the Punjabi WSD using Topical Features ... *200*
7.2.2 Analysis of the Punjabi WSD Based on Local Features ... *201*
7.2.3 Analysis of the Punjabi WSD Based on Syntactic Features ... *202*
7.2.4 Analysis of the Punjabi WSD Based on Semantic Features ... *204*
7.2.5 Analysis of the Word Embedding Features for ML Punjabi WSD ... *204*
7.3 RESULTS OF THE PUNJABI WSD USING DEEP LEARNING CLASSIFIERS ... 208
7.4 TESTING OF THE PUNJABI WSD SYSTEM ... 209
7.5 ERROR ANALYSIS OF THE PUNJABI WSD SYSTEM ... 210
7.5.1 Error Analysis of the ML Punjabi WSD System ... *210*
7.5.2 Error Analysis of the Deep Learning Punjabi WSD System ... *211*
7.6 COMPARISON OF PUNJABI WSD WITH EXISTING SYSTEMS ... 212

CHAPTER 8 : CONCLUSIONS AND FUTURE SCOPE **215**

8.1 **CONCLUSIONS** 215
8.2 **FUTURE SCOPE** 217

LIST OF PUBLICATIONS BY THE AUTHOR **220**

REFERENCES **221**

Chapter 1 : Introduction to Word Sense Disambiguation

Lexical ambiguity is a fundamental characteristic of natural languages. The words of a language can have multiple meanings and are called ambiguous words. Word Sense Disambiguation (WSD) automatically searches for the proper sense of an ambiguous word in a given context from a knowledge resource. In this chapter, the WSD task and its types have been explored. The process of automatic disambiguation of the ambiguous word has explained. The WSD is crucial for various Natural Langauge Processing (NLP) tasks. The challenges for the development of WSD for various NLP have been stated.

The evolution of multilingual computer applications that makes WSD an essential task for their implementations has encouraged us to explore the WSD for the Punjabi language, a prominent South Asian language. The unique attributes of the Punjabi language and its distinctions from the English language have helped us formulate the gap analysis for the Punjabi WSD. The research objectives have been formulated to overcome the gaps in the development of the Punjabi WSD. Finally, the research contributions to achieve the Punjabi WSD have been stated.

1.1 Overview of the WSD

The words of a natural language have a distinct meaning for the context in which they have been used. For example, the word *'newspaper'* in the sentence *'the newspaper got wet in the rain'* is an object that got wet, while in the sentence *'The newspaper fired some of its editing staff',* it refers to an office. Human beings can understand these ambiguities because of their knowledge about the syntactic and semantic structure of the natural language. Machines have to perform some specific tasks that would have helped them understand and interpret natural language ambiguities.

The objective of the WSD task is to automatically select the appropriate meaning/sense of an ambiguous word based on the context of the word from a sense inventory/dictionary. Therefore, WSD is also described as a semantic ambiguity resolver. The mapping of words in context with their corresponding senses coded into a sense inventory can return more than one sense. However, the goal of the WSD is to assign the most suitable sense to the words.

WSD task encompasses two steps. In the first step, different senses for every word related to the text or discourse under consideration are explored. The senses are discovered from the sense inventory. The sense inventory can be the lists of machine-readable dictionaries,

thesaurus, ontologies, WordNet, *etc*. The second step assigns the suitable sense to each occurrence of a word in context. Therefore, the WSD task has been considered the classification task where the senses of the ambiguous word act as classes.

1.1.1 WSD a Classification Task

The WSD task has been described as the classification task where the senses of the words act as different classes, and a classification method assigns the sense to ambiguous words with the help of context and knowledge resources. The classification methods have been used in NLP tasks like text categorization (assigns predefined labels to target text), part-of-speech tagging (assigns the parts of speech tags to target words in context), named entity resolution (assigns predefined categories to target textual items), *etc*. In all these NLP applications, there are fixed classes (parts of speech, categories, *etc*.). However, the WSD task is composed of *n* distinct classification tasks where *n* is the size of the lexicon. The set of classes for the WSD classification task changes depending upon the ambiguous words. The WSD needs the representation of common sense and encyclopaedic knowledge to resolve the sense of an ambiguous word. Thus, the WSD problem is defined as an AI-complete complexity problem (Navigli, 2009).

Formally, the automatic WSD task identifies the correct sense of some ambiguous target word (w_t) or all words (w_1, w_2, …, w_t, …, w_n) present in the context *C*. The *S(i)* is the subset of the word (w_t) senses that are coded in the dictionary *D* as given in equation (1.1). Typically, only the correct sense is selected such that $|S(i)|=1$.

$$S(i) \subseteq SensesD(w_t) \quad \ldots (1.1)$$

1.1.2 Different Types of the WSD

Two different types of WSD classification task are targeted (lexical) WSD and all word WSD.

- **Targeted WSD** resolves the ambiguity of a restricted set of that target ambiguous words, usually occurring one per sentence. This variant has limited coverage. It disambiguates a single target ambiguous word occurring in the sentence. For example,

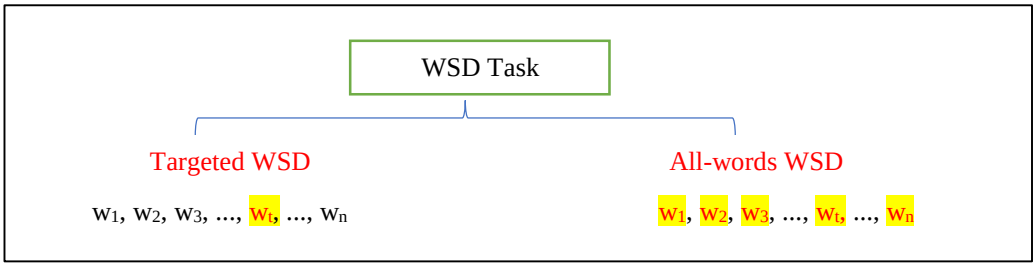


Figure 1.1: Explaining the WSD task

in Figure 1.1 the targeted WSD task disambiguate a single highlighted target word w_t in the tokenized sentence. The targeted WSD has limited scope and is mostly configured as supervised classification tasks (Elayeb, 2019). The supervised tasks have been trained using the labelled dataset, and then these trained models have predicted the sense of an unlabelled target ambiguous word.

- **All-words WSD** task disambiguates all open-class words (noun, adverb, verb, adjectives) in a text. The all-words WSD task considers all the words in the running text or the sentence irrespective of their grammatical open-class, i.e., nouns, verbs, adjectives and adverbs. The all-words WSD task has wide coverage and relies on the coverage of language by the knowledge resources (Navigli, 2009). The highlighted words of the example sentence in Figure 1.1 indicate the target words of all-word WSD task.

WSD can be implemented as an implicit/embedded module or as an explicit/independent module. The implementation methods for WSD are as follows.

(i) **Implicit/Embedded WSD:** The computational identification of the correct meaning of the words in the context is essential and critical for the NLP tasks. Most of the NLP tasks have developed an intrinsic measure to resolve the lexical ambiguity. The development of the specific WSD task within the NLP task is called the an implicit/ embedded WSD. The implicit WSD, as shown in Figure 1.2(i), is domain or task-dependent. Therefore, the overall performance of the NLP task decides the quality of the implicit WSD.

(ii) **Explicit/Independent WSD:** In explicit or stand-alone WSD implementation, the WSD task resolves the ambiguity of words independent of any NLP task. Figure 1.2(ii)

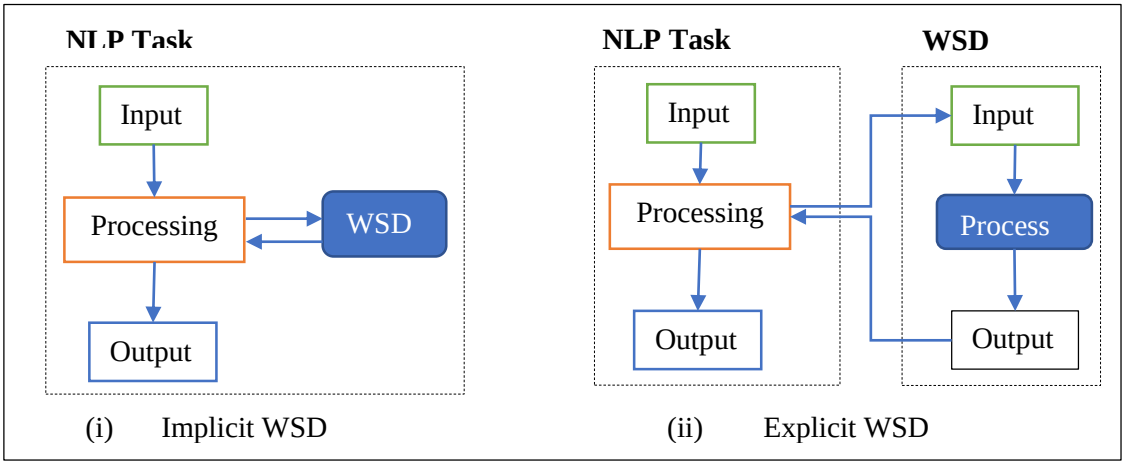


Figure 1.2: Implicit and explicit usage of WSD module

shows the usage of explicit WSD in the NLP task. The explicit WSD can be included in the different NLP tasks externally. The explicit WSD is easier to define, experiment, evaluate and gives broad insight into the computational semantics.

The four main components that have been employed to perform WSD are sense representation, knowledge resources, context representation (features selection), and classification techniques. The implementation process of the WSD task is as follows.

1.2 WSD Process

WSD is a process that maps the sense knowledge of the word (sense vector) with the context information (context vector) of the word to determine the sense of the word. The sense representation is the explanation of the meaning of the word in the dictionary. The dictionary acts as the knowledge resource. The individual words in the context or discourse represent the context vector. Finally, the classification techniques map the sense vector with the context vector to disambiguate the word. Figure 1.3 shows the different steps of the WSD process. The different classification techniques in the WSD process exploit the context and sense vectors differently.

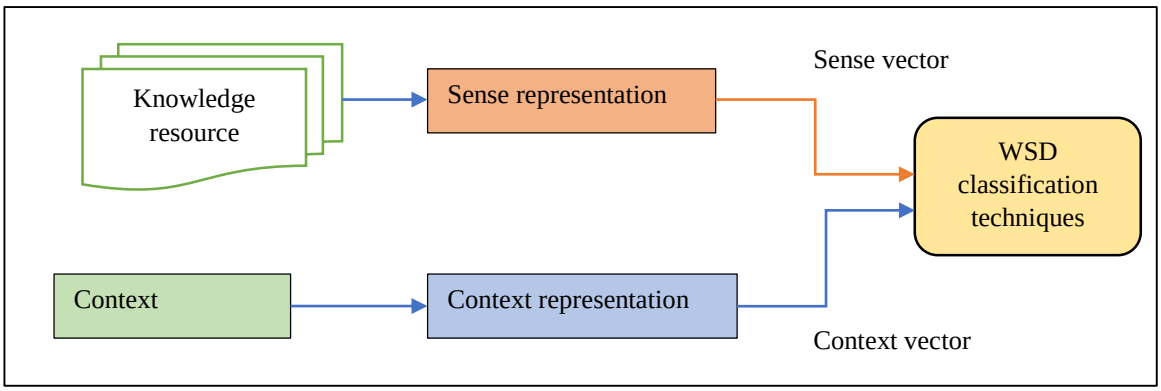


Figure 1.3:The process of word sense disambiguation

The WSD classification techniques can be categorized into supervised, knowledge based and unsupervised approaches. Typically, the targeted supervised WSD approaches follow the steps given in Figure 1.3. The knowledge based approaches exploit the structure of knowledge resources and map them with the target context to determine the senses of all words present in the context. This combinatorial mapping of the knowledge vector with the context vector helps achieve the all-words WSD task. In contrast, the unsupervised techniques of WSD explore the unlabelled knowledge resources. The unsupervised techniques of WSD obtain the different senses of ambiguous words from the input text (context vector) by grouping the word existences.

The details of the individual steps such as the word sense representation, the use of external knowledge resources, context representation, and the classification approaches of WSD are as follows.

1.2.1 Knowledge Resources

The knowledge is the most essential component of the WSD process. The different knowledge resources represent the senses of ambiguous words. These knowledge resources can be distinguished based on the representational format of the word meanings. Figure 1.4 displays the different categories of knowledge resources (structured and unstructured) that provide essential information to associate senses with words. The structured knowledge resources can be distinguised as lexical (Dictionaries, Thesauri, WordNet's) and ontological (ontologies) resources. The lexical resources include linguistic information such as phonetics, syntactic, semantic relations, combinatory behaviour, meanings, *etc*. In comparison, the ontologies are non-linguistic and provide the taxonomy and relationships of the objects in a specific technical domain (G, 2004). Therefore, the ontologies have been examined for domain-specific WSD while the lexical resources have been explored comprehensively in WSD research.

The supervised and knowledge based WSD techniques extract the senses of words from the structured (Dictionaries, Thesauri, WordNet's and Ontologies) resources and assigns them to the ambiguous words. The unstructured resources (Corpora and Word Wide Web text) are commonly explored by the unsupervised techinques of WSD to estimate the senses of the words. The text in unstructured resources is linguistically structured but, it is unstructured from a computational perspective.

1.2.1.1Structured Resources

The structured resources represent the words, relationships and their attributes in an organized manner.

- **Dictionaries:** The Machine Readable Dictionary (MRD) has been used as the primary knowledge source for the different NLP tasks since the inception of this research area. The Collins English Dictionary, the Oxford Advanced Learner's Dictionary of Current English, the Oxford Dictionary of English (Soanes & Stevenson, 2003), and the Longman Dictionary of Contemporary English (LDOCE) have been extensively used dictionaries in NLP tasks. Nonetheless, the evolution of the computational abilities of machines has enhanced the dictionary with different semantic relations among the words. Before the

diffusion of MRD into WordNet, the MRD has extensively applied for WSD (Navigli, 2009).

- **Thesauri:** These are the structured resources that organise the words into different categories. The categories are the relational information among the words like synonym, antonym *etc*. Roget's International (Roget, 1911) and Macquarie Thesaurus (Bernard, 1986) have been commonly used linguistic resources for the English language.

- **Ontologies:** These resources describe the words with concepts. Ontologies are domain-specific, and they include taxonomy and the semantic relations among words. The Unified Medical Language System (UMLS) contains the medical concepts and terminologies. This terminological resource has been explored in medical WSD (Schuemie, et al., 2005; Jimeno-Yepes, et al., 2011)

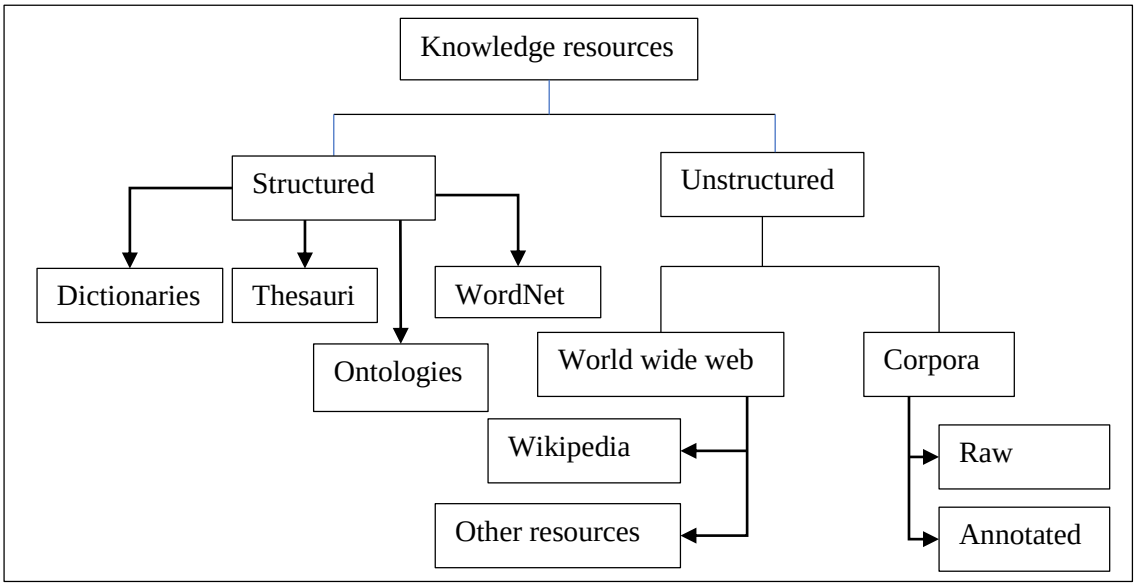


Figure 1.4: Different types of the knowledge resources

- **WordNet:** It is an extensive lexical database of words and their respective meanings. Synonymous words are grouped to form ordered sets called synonym sets or synsets. Word forms with multiple distinct meanings are represented in as many distinct synsets. Each synset represents a single distinct sense or concept. Synsets are interlinked, employing conceptual semantic and lexical relations (Fellbaum, 1998). The structured WordNet lexicon has additional information such as the selectional preferences and frequency information of the senses of the words. WordNet describes the synset with the help of gloss. The gloss of synset contains a brief definition that is one or more short sentences illustrating the use of the synset members. One or two example sentences have also been included to elaborate the usage of synset word. The words in the WordNet have been

classified into noun, verb, adverb and adjective Part of Speech (PoS) categories. The WordNet also consists of compound words. The compound words are the collection of two or words treated as a single word. WordNet defines a variety of semantic and lexical relations between the synsets and words. Semantic relations define a relationship between two synsets. Semantic relations connect the word senses included in the synsets. English WordNet by Princeton University has acted as a de-facto knowledge resource for the implementing of other languages WordNet's such as EuroWordNet (Vossen, 2002), BalkaNet (Tufis, et al., 2004), MultiWordNet (Pianta, et al., 2002), Open multilingual WordNet (Bond & Foster, 2013), and IndoWordNet (Bhattacharyya, 2017), *etc*. A unified multilingual semantic network resource BabelNet, merges the knowledge from Wikipedia, Wiktionary and WordNets (Navigli, et al., 2021). A universal platform Global WordNet Assocaition is available at http://globalwordnet.org/ that enables the users, and researches to share, deliberate and connecting all language WordNet of the world.

The structured knowledge of the sense inventories yields the different meanings of a word and provides meaningful inter-word relational information. This information has been successfully explored by the knowledge base WSD method to determine the sense of ambiguous words.

1.2.1.2 Unstructured Knowledge Resources

These resources are the simple collection of text which can be raw or sense annotated. The supervised and unsupervised approaches of the WSD task have successfully explored these resources to determine the sense of ambiguous words. The details of different unstructured resources are as follows.

- **Word Wide Web (WWW):** The exponential growth of the internet, that is, WWW, has increased the availability of unstructured text resources and their usage in the WSD tasks. Wikipedia a partly structure text that describes all concepts of a language in a single platform available on the WWW. Wikipedia, an open text dataset for different languages, has been effectively used in semi-supervised and unsupervised WSD tasks (Navigli & Ponzetto, 2012; Dandala, et al., 2013).
- **Corpora:** The raw corpus is a collection of text without any sense or word descriptions. The annotated corpus contains the senses of the words in the context. The supervised WSD approaches explore the annotated corpus, while the unsupervised approaches use the raw corpus to disambiguate the ambiguous word in context.

The WSD has also investigated the other language resources such as word frequency lists, acronym lists, stop word lists, and domain labels.

1.2.2 Sense Representation

The representation of the different senses of the words is an important step of the WSD process. The most suitable meaning of a word is called its sense. The selection of the sense of a word is variable, context sensitive and application dependent. The clear distinction among the sense definitions of the words is difficult (Ide & Wilks, 2007). The senses of the words are overlapping, which means there exists no clear demarcation between the two senses of words. Due to such uncertainties, the dictionaries will consist of different choices of senses for the same words.

The meaning or sense of words are represented in sense inventories. The encoding of discrete senses of ambiguous words in the sense of inventory is complex because the natural language inherently changes. WSD task uses the sense inventories to interprets the distinct sense of the words. The enumerative and the generative approaches have been used to represent senses of the words in the dictionaries (Ide & Wilks, 2007). To perform WSD task, the sense representation in the machine-readable dictionaries must embrace the enumerative approach commonly used in paper dictionaries (Navigli, 2009). The enumerative approach represents the sense distinction in the sense inventories. However, the enumerative approach limits the dynamic expressivity of a word within a predetermined set of senses (Kilgarriff, 1997; Kilgarriff, 2007). Further, the enumerative approach also has issues like the organisation of senses, the distinction between fine and coarse senses, and distinguishing of senses based on attestations of collected text.

A generative approach expresses the senses of the words in qualia roles (i.e. the semantic features which capture knowledge about the entity), and it has tried to overcome the issues of the enumerative approach (Navigli, 2009). Other approaches of sense distinctions like sense inductions from text and linguistic criteria have also been investigated for sense representations. Nevertheless, for objective evaluation and comparison of the WSD task, researchers have utilised the sense inventories based on the enumerative approach.

Moreover, the sense selection is dependent on the NLP tasks. In some applications, abstract senses (coarse senses) are sufficient, while others NLP tasks require elaborated sense definitions (fine sense). Sense granularity (coarse and fine sense) affects the disambiguation process. The fine overlapping senses of words are difficult to interpret by the WSD, while the

coarse senses may miss the critical sense. However, (Ide & Wilks, 2007) have shown evidence that the assignment of fine sense by humans and machines is difficult while they perform exceedingly well to assign the coarse level senses. Psycholinguistics plays a crucial role in sense definitions (Kilgarriff, 2007). Only after sense representation and selection the ambiguities such as morphological, syntactic, lexical, semantic and pragmatic can be observed. The representation of the context which contains the ambiguous word is as follows.

1.2.3 Context Representation

The individual words of the text or discourse form the context of the ambiguous word. The context has been processed and transformed into a structured form. The machine learning classifiers process this structured information for the disambiguation of ambiguous words. Different steps generate structured contextual information, namely tokenisation, PoS tagging, lemmatisation, chunking, and parsing.

The context of the ambiguous word provides the right clue about the sense of the word. The word ambiguities can be due to morphological, syntactic, lexical, semantic and pragmatic ambiguities (Bhala & Abirami, 2014). Contextual features are selected from the context of the ambiguous words to resolve the ambiguity of words. Table 1.1 shows the different contextual information used for the WSD process.

Table 1.1: Different types of features based on the context representation

Features	**Description**	**Representation**
Local features	The immediate neighbouring words of a target word	PoS tags, collocations
Topical features	Document of general discussion	Key topical words
Syntactic features	Morphological characteristics of words within some window size	Normal forms, parse trees and dependency graphs
Semantic features	The words distributed in the feature space and their similarity relations	Domain specific words, sectional restrictions

The local features describe the relation of the target word with the neighbourhood words. The topical features describe the global meaning of words based on the topic. Syntactic features have described the grammatical relationship between different classes of words like verbs, nouns, adjectives and adverbs. The semantic information is extracted from a knowledge base like WordNet that establishes the relations between the different concepts of the words. Supervised classification approaches primarily use the flat structured representation of context. The flat nature of the context representation is due to the simple structure of the training dataset. While the knowledge based and unsupervised context approaches need highly structured representations like trees and graph structures. Tree and graph structures represent the

relational concepts encoded in semantic networks and computational lexicons (Agirre, et al., 2014). The performance of the WSD system is sensitive to context size irrespective of the supervised or unsupervised approaches.

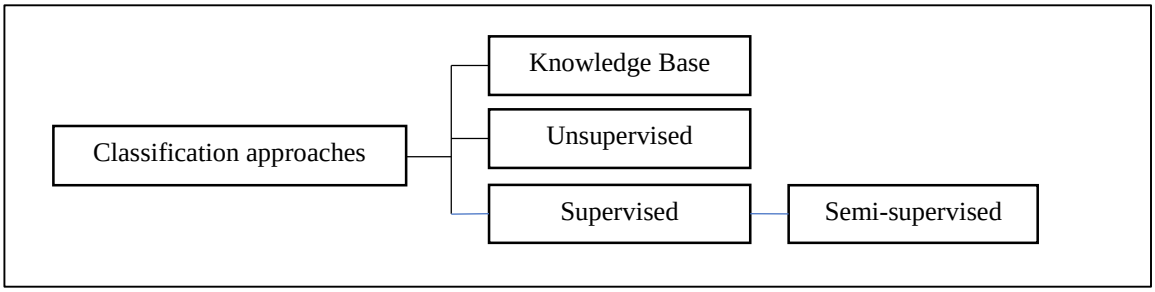


Figure 1.5: Classification approaches used for WSD

1.2.4 WSD Classification Techniques

The final step of the WSD process is the selecting the most suitable sense of the ambiguous words by a classification technique. The WSD classification techniques identify the most suitable sense of the ambiguous word in a sentence by exploring the sense vector extracted from the external knowledge resources and the context vector selected from the context of the ambiguous word. The different classification approaches are shown in Figure 1.5 that have explored the sense knowledge and contextual representation of text to disambiguate the words. The classification approaches can be broadly distinguished into the knowledge base, unsupervised, supervised and semi-supervised approaches.

1.2.4.1 Knowledge Base Approaches

These approaches use external resources such as machine-readable dictionaries, thesauri, glossaries, ontologies, WordNet, *etc*., for the sense disambiguation. The first knowledge base approach by (Lesk, 1986) has explored a machine-readable dictionary to determine the sense of the word. The innovation of the WordNet of the English language, a semantically rich database, has enhanced the performance of the knowledge base approaches (Agirre, et al., 2006). Knowledge base approaches have been mainly categorised into selectional preferences, overlap and structural algorithms on the basis of information selection procedures from the different knowledge resources (Navigli, 2009). The taxonomy of the knowledge base algorithms is shown in Figure 1.6.

- **Selectional preferences algorithm** explores the word to word or word to class semantic or syntactic relations to select only those senses that are in harmony with relations. The application of selectional preference for the English WSD has yielded almost as good results as Lesk-based and the most significant baseline classifiers (Mihalcea, 2007).

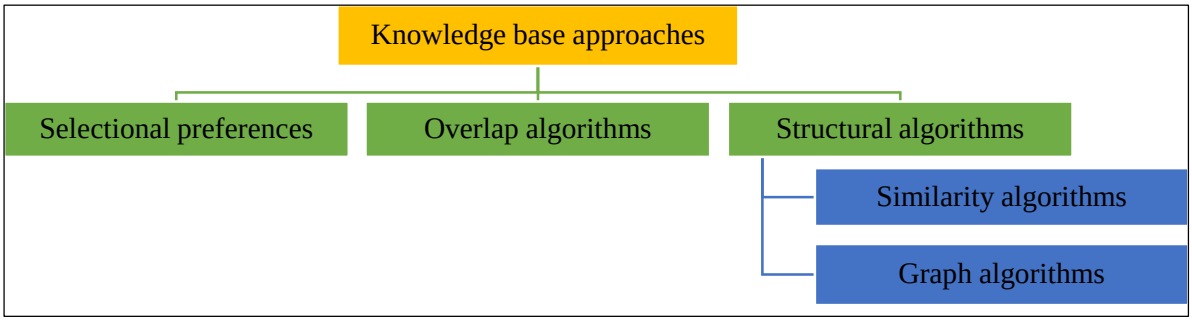


Figure 1.6: Types of knowledge base approaches of WSD

- **Overlap algorithm** compares the gloss of every sense of the target word in the phrase with the gloss of every other word in the phrase. The sense whose gloss has the highest overlap with the glosses of every other word in the phrase is selected as the winner sense of the target word in the context. (Lesk, 1986) disambiguates all the words in the context by comparing the dictionary definitions of senses of all context words. This lead to the sense combinational explosion problem (Vasilescu, et al., 2004). To overcome it, a simplified overlap algorithm has been applied to identify the correct sense for one word at a time rather than classifying all words at the same time. (Banerjee & Pedersen, 2003) have used the extended gloss overlap method with the WordNet such that overlap of glosses of words semantically related with target word has also considered. (Mihalcea, 2007) has explored the corpus-based Lesk algorithm, which augments the sense tagged dataset of an individual ambiguous word to classify the target word. (Schwab, et al., 2013) have applied the Ant colony algorithm to the Lesk algorithm and have reported an improvement in the disambiguation task. (Basile, et al., 2014) have modified the Lesk algorithm, which computes the gloss-context overlap for a word with the help of a similarity function defined on a distributional semantic space. (Oele & Van Noord, 2017) have exploited the word embedding over the Lesk algorithm to compute the similarity between the gloss of a sense and the context.
- **Structural algorithm** exploits the structure of the computational lexicons like WordNet to disambiguate the sense of the ambiguous word. There are two main kinds of structural algorithms *similarity-based*, and *graph-based*, that explore the structure of computational lexicons.
  - ***Similarity-based*** algorithms compute semantic similarity over the network of semantic connections between word senses. The semantic similarity measures algorithm deliberates the local context along with the syntactic relations. These methods do not include additional contextual information beyond a certain window

size. Words that share a common context are closely related in meaning, and therefore the appropriate senses can be selected by choosing those meanings found within the smallest semantic distance. (Pedersen, et al., 2004) has developed the English WordNet similarity package that has explored the six semantic similarity measure between a pair of concepts. (Navigli & Velardi, 2005) have created the Structural Semantic Interconnections (SSI), a structure of different sense of the ambiguous word. The authors have exploited the SSI relations according to the grammar to select the most suitable sense. (McCarthy, 2006) has applied the similarity measure on the words of the English WordNet to produce sense clusters that can be varied according to the needs of the applications. (Patwardhan & Pedersen, 2006) have developed a semantic similarity measure by exploring the co-occurrence information along with the content of the WordNet. (Ponzetto & Navigli, 2010) have extended the WordNet with Wikipedia by mapping the semantic relations between the two resources. The authors have estimated the conditional probability to find the best sense. (Miller, et al., 2012) have applied lexical expansions derived from a large corpus using dependency-context similarities. (Garla & Brandt, 2013) have derived semantic similarities from the Unified Medical Language System (UMLS) to perform WSD. (Wang & Hirst, 2014) have applied the NB models to determine the probabilistic similarities of overlapping gloss to find the most suitable sense.

- ***Graph-based*** algorithms have exploited the graph structure of the computational lexicon to determine the appropriate sense of the word in the context. The nodes of the graph are the words of context, while the weighted edges of the graph are the semantic relations among nodes. During the disambiguation, a score is calculated by adding all connections weight involving a sense. The sense with the highest score is declared as the winner sense. (Chen, et al., 2010) have applied the WordNet as the sense determination source for the graph and dependency tree created from the raw corpus. (Agirre, et al., 2014) have applied the Random walks graph-based method over the large lexical knowledge bases. (Wang & Hirst, 2014) have explored topical and semantic associations. The authors have created the graphs from the semantic and context text of the target words to disambiguate the ambiguous words. (Chaplot, et al., 2015) have created a web interface that determines the senses of all words present in the query instance using the Markov Random Field (MRF) built from the WordNet and a dependency parser. (Moro, et al., 2014) applied a unified approach based on high-performance graph based Personalised PageRank algorithm on the BabelNet (a

multilingual semantic network) for multilingual entity linking and WSD. (Ding, et al., 2021) have designed a dynamic graph structure that reduces error propagation and successfully evaluated it on intent detection and slot filling tasks.

The advantage of knowledge base approaches is their coverage of the language. This is due to the scale of the knowledge resources.

1.2.4.2 Unsupervised approaches

These approaches are based on raw or unlabelled corpora. These approaches do not involve human efforts and external resources like a dictionary, ontologies or a thesaurus, *etc*., to identify the different word senses of the ambiguous word. These approaches are based on the idea that the same sense of a word will have similar neighbouring words. Unsupervised disambiguation generally clusters the raw text into separate groups. Hence, these approaches are also known as sense discrimination approaches (Navigli, 2009). The assignment of the senses to the clusters need human intervention or employing the clusters in end-to-end applications. These approaches are data-driven and language independent and rely on the distributional characteristics of unannotated raw corpora. The unsupervised approaches can be further classified into translational, discriminative and co-occurrence graph types, as shown in Figure 1.7.

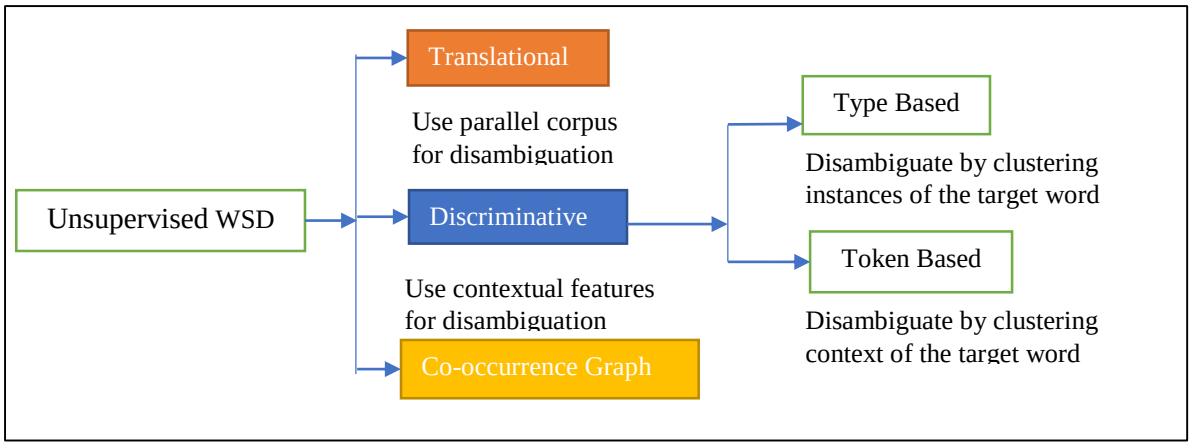


Figure 1.7: Types of unsupervised approaches of WSD

(i) **The translational equivalence approaches** leverage parallel corpus of different languages for disambiguation. It is based on the fact that different senses of the words in the source language can be translated into different words in the target language. This approach requires word-aligned parallel corpus; that is, source words should be translated to target words. The word alignment and the availability of the large quantity and quality of the

parallel text are the major challenges of the translational approach. The translational unsupervised WSD method has been integrated and evaluated into the translational applications (Pedersen, 2007; Kumar & Yadav, 2018).

(ii) **Discriminative unsupervised approaches** use the monolingual corpora. These approaches are based on the characteristics of the occurrence of similar words in a similar context (Pedersen, 2007). Discriminative approaches can be further classified into the type and token-based methods.

a) **The type-based methods** identify the related set of words from the context. The context in type-based methods is represented as a word co-occurrence matrix. These matrices are based on the count of word occurrence or log-likelihood or pointwise mutual information association measures (Pedersen, 2007). The co-occurrence matrix has produced the feature vectors for each word in the corpus. The different algorithms that are used to get the related words from co-occurrence features are as follows.

- Latent Semantic Analysis(LSA) is based on the co-occurrence matrix of the word in the context. In LSA context is a sentence, paragraph or document. The dimensionality of word by context co-occurrence matrix of corpus words is enormous. The LSA handles big dimensionality problems by the application of SVD (singular value decomposition). SVD transforms the large matrix into a smaller one by collapsing a similar context. The similarity among the word vectors formed is estimated by the cosine angle between two vectors. (Van de Cruys & Apidianaki, 2011) have applied LSA for creating sense inventory by inducing senses from raw corpus which has been used for the disambiguation of ambiguous English words. (Martin, 2018) has explored the LSA for word sense induction and disambiguation and has evaluated the five ambiguous English words.
- Hyperspace Analogue to Language (HAL) uses the word by word representation of the corpus words in a co-occurrence matrix. The word co-occurrence is dependent on word order. Each word in a co-occurrence matrix is represented by a row and column. The word is finally represented by a single vector that is a concatenation of its row and column vector. The high dimensionality is handled by Multidimensional Scaling (MDS). The similarity among the word vectors formed is measured by calculating the Euclidean distance between two vectors (Pedersen, 2007).
- Clustering by committee (CBS) approach finds synonyms for each meaning of a word. In this approach, similar words (synonyms) to the target word are identified, and

clusters are formed. It does not conflate the all meaning of a word into a single representation. The CBS is a three-step algorithm. The co-occurring matrix of the words is created in the first step that uses the Pointwise Mutual Information (PMI) between a word and a particular context. The context of the word consists of two words and their syntactic dependencies (subject-verb, verb-object, adjective-noun, *etc*.). The second step of the algorithm processes the top *k* (between 10 and 20) words associated with the target word found by sorting the PMI values. For each *k* word, CBS extracts the most similar words from the co-occurring matrix and cluster them using average link clustering. The similarity score is assigned to the discovered cluster, and the words in the most similar cluster form a committee. The committee represents the different senses of the target word (Navigli, 2009).

b) **The token-based method** forms different clusters of the context of a target word. The context similarity is estimated between the context of the target ambiguous word. The yielded clusters do not have labels. The input for the token-based methods is different contexts of the target word without any sense label. The context group discrimination algorithm represents entire contexts of target word rather than single word types using second-order co-occurrences of lexical features like unigram and bigram. The agglomerative clustering algorithm explores the local morphology; co-occurrence features to prepare clusters. Context group discrimination uses LSA to yield clusters, while the agglomerative clustering algorithm applies McQuitty's similarity analysis (merge context with the lowest similarity) to yield the clusters (Pedersen, 2007). (Purandare & Pedersen, 2004) have explored both the techniques using the Senseval-2 English lexical sample data. (Manandhar, et al., 2010) have explained the SemEval 14 task in which senses of 100 target words have been induced using the web corpus and disambiguate the test instances using the induced senses. (Butnaru, et al., 2017) have explored the shotgun sequencing technique to disambiguate the ambiguous words and have reported better performance than the state-of-the-art unsupervised WSD algorithms.

(iii) **Co-occurrence Graph approach** leverages the graph data structure. In the first step, a graph $G= (V, E)$ structure has built from the context of the target word. The words act as the nodes V, and an edge E exists between words if they co-occur in context. The weights are assigned to edges of the graph relative to the co-occurrence frequency calculated using equation (2.1) where $P(w_i| w_j) = ferq_{ij}/freq_j$. The weights of the edges between high

occurring words are negligible, while the weights of edges between sparsely occurring words are high. Edges with a weight above a certain threshold are discarded.

$$w_{ij} = 1 - max\{P(w_i|\, w_j)\,, P(w_j|\, w_i\,)\} \quad \ldots (2.1)$$

In the second step, an iterative algorithm is applied to find the node with the highest relative degree in the graph. The set of hubs selected from the senses of the word. The target word and the hubs are connected with zero weight edges, and the Minimum Spanning Tree (MST) of the entire graph is used to estimate the sense of instance. (Veronis, 2004) has explored this Hyperlex approach. (Agirre, et al., 2006) have applied the Hyperlex and PageRank algorithms for the lexical sample and the all word WSD. The authors have stated that graph-based techniques combine both the local and global co-occurrence information for disambiguation, while the clustering approach relies on the local co-occurrence information. (Agirre, et al., 2009) have tuned the adjacent vertices in a hub, the minimum frequencies of edges, vertices, hubs parameters of the PageRank algorithm to perform the unsupervised WSD. (Agirre, et al., 2010) have implemented WSD for biomedical documents for all words present in UMLS Meta-thesaurus. The authors have used the personalized PageRank algorithm on the graph created from the Meta-thesaurus tables to select the best sense for each ambiguous word.

1.2.4.3 Supervised Approaches

Supervised approaches are commonly known as machine learning techniques that induce the rules for classification from the sense tagged dataset. The classifiers in supervised approaches are called word experts because they target a single ambiguous word. These classifiers learn from the sets of examples encoded into the number of features together with their appropriate sense label. Labelled data is prepared manually by referencing the machine-readable dictionaries. Supervised approaches have yielded better results than the unsupervised and knowledge based approaches (Bhala & Abirami, 2014). The classifiers applied for supervised approaches can be segregated on the basis of their approach to process the features selected from the annotated dataset (Màrquez, et al., 2007).

(i) **Probabilistic approaches** explore the distribution of the conditional probability of the target word and its context. The basic probabilistic classifier Naive Bayes (NB) classifier, based on the Bayes theorem, has mostly experimented with the WSD (Navigli, 2009). Multinomial NB takes into account the word count, and it has also been applied for the WSD. (Cai, et al., 2007) have explored the NB and incorporated syntactic patterns, PoS

categories, collocations and single word neighbourhood features to perform the WSD. The Maximum Entropy approach is similar to the NB, but it does not consider the independence of the words assumption. It combines statistical evidence from many resources. At the same time, it strives to satisfy the constraints specified and maximize the entropy. (Su`arez & Palomar, 2002) have presented a maximum entropy-based WSD system that has been analysed on SENSEVAL-2 lexical Spanish data. (Tratz, et al., 2007) have developed a WSD system using a maximum entropy classifier which has produced the highest F-score for the fined-grained English all-words subtask of SEMEVAL.

(ii) **Discriminating rule approaches** apply the selective rules associated with each word sense to determine the right senses. A decision list is the collection of weighted if-then-else rules of the form (condition, class, weight) prepared from the training corpus. For a test example, a decision list sequentially parses the conditional rules and assigns the sense that machetes condition. (Yarowsky, 2000) has applied hierarchical decision lists to perform the WSD. A Decision Tree (DT) is a divide and conquer approach that derives a tree-like structure from the training dataset. Internal nodes of the DT represent a rule, while the leaf node represents the class (Navigli, 2009). (Mohammad & Pedersen, 2004a) have implemented the English WSD using the DT classifier based on the lexical and syntactic features.

(iii) **Exemplar or Instance based approaches** perform disambiguation with the help of all the examples in the memory. k-Nearest Neighbour (k-NN) algorithm finds the sense of the test instance by searching the *k* most similar examples in the memory (Màrquez, et al., 2007). The exemplar-based classifier has produced state-of-the-art performance in WSD (Decadt, et al., 2004).

(iv) **Support vector machines** apply a hyperplane or decision plane to classify the instances. The aim of the Support Vector Machine (SVM) is to find a hyperplane that clearly distinguishes two-class dataset. To handle the multiclass WSD task, SVM applies a one vs all sense approach (Navigli, 2009). The SVM has shown a significant improvement in the WSD task (Zhong & Ng, 2010). (Taghipour & Ng, 2015; Iacobacci, et al., 2015) have experimented with the neural network based word embedding features on the SVM classifier to perform the English WSD.

(v) **Ensemble based approaches** combine the knowledge of different ML classifiers to build a robust single classifier for the classification task. Bagging, boosting, and stacking are the three different ensemble approaches. In the bagging approach, the homogenous classifier is trained on the small training datasets of the same size selected randomly from the domain training dataset. The outputs of these classifiers have been combined using different voting

and averaging deterministic approaches. (Mohammad & Pedersen, 2004b) have applied a bagged DT for the WSD task created from the pairs of lexical and syntactic features. (Narouei, et al., 2015) have applied Random Forest classifiers on sequential patterns in a sentence and have reported improved performance. The boosting approach trains the homogenous classifier adaptively using a weighted training dataset. The boosting approach is iterative such that the new model is influenced by the performance of those built previously. The boosting approach builds a new model iteratively by increasing the weights of the wrongly classified instances of the training dataset by the previous model (Màrquez, et al., 2007). The boosting approach has been applied for the WSD by (Escudero, et al., 2000). The addition of syntactic to traditional features has improved the performance of the AdaBoost classifier for the WSD (Mart'nez, et al., 2002). The stacking approach combines the outcomes of the heterogeneous classifiers using a meta-learning classifier. Ensemble methods have produced state-of-the-art results in the Netflix and Kaggle machine learning competitions (Zhou, 2012). Ensemble methods have also been applied on WSD tasks in SENSEVAL/SEMEVAL competitions, and AdaBoost (Xiu, et al., 2017), majority vote (Mihalcea, 2004), and Stacking (Brody, et al., 2006) ensemble methods have shown an improvement in the prediction accuracy.

(vi) **Neural network** is a connectionist approach that consists of an interconnected group of artificial neurons. Neural Network (NN) supervised approach uses a pair of input features and desired responses as the output. The input, hidden, and output are the three layers of the NN. The NN aims to adjust the weights progressively so that the output unit representing the desired response has a larger activation than any other output unit for every training example. (Véronis & Ide, 1990) have derived a large scale NN from the machine-readable dictionaries to perform the WSD task. The authors have prepared the shallow network that involves all words and their corresponding senses as input and output layers of the NN. (Mooney, 1996) has applied the simple NN without hidden layer for the WSD of the "*line*" ambiguous word, and his system has outperformed the logistic methods. (Zuppann, 2003) has experimented with a simple feed forward NN with no hidden layer on the five English words, and his system has produced a comparable performance to statistical methods. (Azzini, et al., 2008) has devised the encoding schemes for English language text that has been used as input to neuro evolutionary algorithm to perform the supervised WSD. (Lu, et al., 2004) have applied NN for the Chinese WSD.

(vii) **Deep Neural networks** have an additional number of hidden layers as compared to simple NN. The number of hidden layers in the NN decides the deepness of the NN. The

performance of the NN models has been improved by increasing the number of layers of neural networks (Collobert & Weston, 2008). The deep NN capture the quality features directly from the input data. Therefore, the deep NN are called domain-independent or can handle differnet tasks effectively. However, the effectiveness of deep NNs is dependent on the size of input data, and if a sufficient amount of data is given, they outperformed the conventional machine learning approaches (Popov, 2018).

The embedding features devised with the help of a single layer NN have been used by the different deep neural networks to perform the WSD. Word embedding features represent words in a low dimensional continuous space. Word embedding features can capture the semantic features from the massive amount of contextual information.

The simple feed forward NN has been designed by (Bengio, et al., 2003) that predict a word given the preceding words. (Collobert & Weston, 2008) have presented a deeper NN consisting of several layers for feature extraction to build a general architecture for NLP tasks. But the Word2Vec algorithm by (Mikolov, et al., 2013) based on shallow and simple NN has produced the word embedding that has transformed the perspective of the NN for the NLP tasks. (Pennington, et al., 2014) have explored another word embedding model that produces explicit representation from the statistical calculation on word counting. (Bojanowski, et al., 2017) have extended the Word2Vec algorithm by taking into account the sub word information.

(Iacobacci, et al., 2016) have investigated the role of Word2Vec, C&W and Retrofitting Word2Vec word embedding representations in the supervised WSD system. (Faruqui, et al., 2015) have refined the word embedding features by exploiting the semantic knowledge from the paraphrase database. (Rothe & Schütze, 2015; Iacobacci, et al., 2015) have examined the representation of word senses instead of individual words. Word embedding vectors have been successfully applied by the researchers in the WSD by averaging or concatenation of word embedding for each word in a text. But this causes loss of the sequential and syntactic information of the text (Yuan, et al., 2016). The development and application of deep neural network approaches using text or word embedding features are as follows

a) **Multilayer perceptron** consists of more than one hidden layer. These are also called deep feed forward networks because they are made up of more than two neural layers and does not consist of any feedback in the output. The input unit of feed forward network is word vectors extracted from the context of the sense tagged instances. The senses/classes of the training examples specify the output unit.

b) **Long Short Term Memory (LSTM)** models are a type of gated Recurrent Neural Networks (RNN) which do not have a long-term dependency problem, and these also propagate data backwards. The chain-like structure of LSTMs consists of repeating module with a different structure. (Yuan, et al., 2016) have studied the WSD with a sequence learning neural net, LSTM. (Melamud, et al., 2016) have trained a bidirectional Long Short-Term Memory (BLSTM) using unannotated data. (Raganato, et al., 2017) have presented a BLSTM with attention and two auxiliary loss functions (part-of-speech and the WordNet coarse-grained semantic labels).

c) **Transformer based models** explore the language models to perform WSD. The transformers are based on the attention-based neural architectures. Extensive pre-trained Bidirectional Encoder Representations for Transformers (BERT-based) models have improved WSD task performance (Vial, et al., 2019). (Bevilacqua & Navigli, 2019) reported the state-of-the-art result of the WSD task using a Quasi-BERT transformer-based architecture for contextualized embeddings. The transformer-based language models BERT and ALBERT have yielded benchmark results for the WSD task (Loureiro, et al., 2021). The pre-trained language models have yielded state of the art result for different NLP tasks, including WSD. However, they need access to enough training data of resource-scarce languages (like the Punjabi language) and computing power to produce efficient language models. WSD process is an arduous process that involves the creation of high-quality complete coverage training data, focusing on the verbs along with nouns and collecting specific knowledge from lexical resources.

(viii) **Semi-supervised approaches** train the classifiers from a small set of sense tagged dataset and, thus they need minimal human intervention. The semi-supervised approaches try to overcome the problem of knowledge acquisition in supervised approaches. The main semi-supervised approaches applied for the WSD are bootstrapping and monosemous relatives' approaches.

a) **In Bootstrapping approach,** classifier *L*, with the help of annotated seed instances *S* labels, the unlabelled dataset *C* and merges the output *O* (labelled suitable instances) into the annotated dataset ($S = S+O$). This iterative process increases the size of the annotated dataset and shrinks the unlabelled corpus. There are two self-training and co-training types of bootstrapping approaches applied for the WSD (Mihalcea, 2004). The self-training bootstrapping applies a single classifier at each retraining of its output, while the co-training applies two classifiers for each retraining. The selection of optimal values for parameters like iteration size, confident value for selection of unlabelled data

to be added to the labelled dataset, and pool size are the major impediments for self-training and co-training approaches (Navigli, 2009).

b) **In the monosemous relatives** (synonym words with unique meaning) approach, the seed dataset is applied to the enormous web data to constitute the sense tagged dataset. The unique expression for a sense *S* of word *w* has been prepared from the sense inventory. These expressions are compared with the web data, and similar text snippets are selected. The collected test snippets constitute the sense tagged dataset for a sense of the word (Navigli, 2009). The use of web data can alleviate the problem of data sparseness in training sets.

Identifying a sense of an ambiguous word depends on the context, knowledge resource and the classification technique used. The applications of the WSD task are discussed in the following subsection.

1.3 Applications of WSD

The ambiguity is the primary concern in almost all the NLP tasks. Thus, the WSD can play an important role in the different NLP tasks. The goal of sense disambiguation depends on the NLP tasks. For instance, a speech synthesis requires homograph disambiguation words. The homograph words have different meanings with same spelling. However, they have the same or different pronunciation (e.g "*Advocate*" (means "*write in support of*" and pronounced with a long "*a*" sound) / "*Advocate*" (means "*leads the cause of another*" and pronounced with a short "*a*" sound). Similarly, the speech recognition application requires homophone disambiguation words. The homophone words have different sense, different or same spellings, but identical pronunciation (e.g '*cell*' / '*sell*'). Likewise, the written language processing applications might require polysemy disambiguation (polysemy words have the same spelling and pronunciation but have different meanings). Different NLP tasks that are using WSD implicitly are as follows.

(i) **Machine Translation:** WSD is considered the main task to be solved to enable machine translation (Navigli, 2009). A machine translation system must contend with two types of lexical semantic ambiguity. First, there is an ambiguity in the source language where the meaning of a word is not immediately apparent. Secondly, ambiguity in the target language, when a word is not ambiguous in the source language, but it has two possible translations. Hence, in resolving sense ambiguity, a WSD system will help a machine translation system to determine the correct translation for an ambiguous word. The intial research of combining of machine translation with WSD showed contrasting evident (Carpuat and Wu

2005). However, recently researchers have shown that the WSD can enhance the performance of machine translation systems. The neural machine translation models using sense embedding by (Gonzales, et al., 2017), statistical machine translation using WordNet by (Vintar & Fišer, 2016) and adaptive sense clusters by (Pu, et al., 2017) for sense disambiguation have shown the enhanced performance for different bilingual translations. However, most of the machine translation models do not have explored the explicit use of WSD.

(ii) **Information Retrieval (IR):** It relies on the context typed by the user in the query to retrieve documents relevant to the intended sense. For example, the query for "*Java*" would return the document like a type of "*coffee*", "*programming language*", and "*a location*". It is desirable that IR system should return the documents for the proper sense of the "*Java*" word. The usefulness of WSD in the IR field has been shown by (Chifu & Hristea, 2018). (Basu, et al., 2018) have reported an improvement in IR and text understanding by applying WordNet context and additional information like co-occurring terms, phrases, and Wikipedia entities. (Hristea & Colhon, 2020) have reported an improvement in the IR system results using the unsupervised WSD. The query expansion approach by (Hassan & Hadi, 2017) has improved the IR using the fuzzy logic WSD. (Rais & Lachkar, 2018), have improved the Biomedical IR system by applying the conceptualisation and disambiguation strategies in the indexing process. The performance of the query-based text summarization method that extracts main sentences from text documents according to the query has been improved by including WSD (Rahman & Borah, 2020).

(iii) **Information Extraction and Text Mining:** Accurate analysis of the text will improve significantly using the WSD system. For example, bioinformatics research requires the relationships between genes and gene products to be catalogued from the vast scientific literature; however, genes and their proteins often have the same name. An intelligence-gathering system might require the flagging of illegal drugs rather than medical drugs. Acronym expansion, i.e., "*MG*" as magnesium or milligram, can be resolved by the WSD system. The WSD system in named entity recognition would also enhance its coverage and performance (Navigli, 2009). (Andrea Moro, Alessandro Raganato, Roberto Navigli, 2014) have developed a unified graph approach (Babelfy http://babelfy.org) to implement entity linking and WSD for different langauges. (Andrea Moro and Roberto Navigli 2015) have investigated similar means of methods for the entity linking and WSD using BabelNet 2.5.1.

(iv) **Word Processing:** The WSD can help in correcting the spelling of a word. The WSD aids in case change or determination and correction of diacritics for the word in the text (Elayeb, 2019).

(v) **Lexicography:** The WSD and lexicography can complement each other (Kwong, 2012). A lexicographer can provide improved sense inventories, and sense annotated corpora that can benefit WSD. In contrast, the WSD can provide empirical sense groupings and statistically significant indicators of context for new or existing senses to the lexicographers. The WSD can help in creating semantic networks out of machine-readable dictionaries.

(vi) **Content Analysis:** It analyses of the general content of a text in terms of its ideas or themes. Content analysis can certainly be benefited from the application of sense disambiguation. The WSD helps proper classifications of data in blogs and social networks. (Akkaya, et al., 2009) have performed the subjectivity analysis (extraction of opinions, emotions, and sentiments in text) of the text using WSD.

(vii) **Semantic Web:** The WSD can enrich large-scale semantic networks. The domain-oriented WSD would benefit semantic networks in their creation by enabling interchangeability among systems, ontologies and users (Navigli, 2009). SyntagNet, a novel resource consisting of manually disambiguated lexical-semantic combinations, has enabled the knowledge-based WSD systems to outperform the supervised WSD systems (Maru, et al., 2019).

(viii) **Sentiment analysis:** WSD has improved the prediction of the tweet's sentiment from Twitter and the Short Message Service (SMS) messages. (Hung & Chen, 2016) have applied the WSD techniques to build WSD-based SentiWordNet lexicons and have observed an improvement in sentiment analysis using WSD-based SentiWordNet. (Pamungkas & Putri, 2017) have reported improved performance of the lexicon-based sentiment analysis to disambiguate ambiguous words using path-based semantic relatedness. (Seifollahi & Shajari, 2019) have explored the application of the WSD system in the sentiment analysis of news headlines to predict the FOREX market and have reported an improvement in the determination of the sentiment.

(ix) **Emoji word sense disambiguation:** Emojis are pictorial symbols used in informal communication channels like electronic messages and Web pages. WSD has enabled the prediction of Emoji more congruent with the intended semantic meaning of the corresponding trigger word (Bellegarda & Barman, 2019).

(x) **WSD for Sign Language:** An automatic translation of public access big data resources to sign language must be as accurate and free of ambiguities as possible. (Naranjo-Zeledón, et al., 2019) have elucidated the techniques of WSD for sign language and have shown that it can enrich the sign language concepts.

The lexical ambiguity is critical for NLP tasks, and it has been tackled using WSD methods. The sense resolution has improved the performance of every NLP task. Nevertheless, the usage of the WSD system in various NLP tasks is computationally intensive and complex. Consequently, the development and implementation of accurate WSD are essential since different NLP tasks' performance depends on the WSD task. The challenges in the development and implementation of WSD are discussed in the following section.

1.4 Challenges in Development of WSD System

The WSD can benefit the NLP tasks. However, there are different challenges in automatic sense disambiguation that should be addressed to obtain an accurate WSD. Although sense disambiguation comes naturally to human beings, the WSD is an AI-complete problem, which means that the true WSD can be achieved when computers become as intelligent as humans. Some of the challenges faced in computational sense disambiguation are as follows.

(i) The knowledge about the different senses of the words is the primary component of the WSD. A computer needs a sense inventory to understand the different senses of a word. The computer searches the sense inventory to find the right sense of the word. The sense inventories are not consistent in their design. Different dictionaries and thesauruses will provide divergent divisions of words into senses. The non-existence of standard sense inventory is a fundamental challenge in the WSD system.

(ii) Comprehensive dictionaries and corpora are indispensable for the effectiveness of WSD. Recent developments in computational facilities have led to the creation of much better MRDs like WordNet, which is suitable for written language understanding. WordNet resources describe the meaning of the words and provide the concept definition, example sentences, and different relationships among the synonym concepts.

(iii) The creation of knowledge resources is a human-intensive and expensive task. The existing knowledge bases are limited, and the resulting disambiguation systems can handle only a handful of words from a simplified domain.

(iv) Selection of type of word sense, i.e., fine or coarse-grained sense of target word, is an arduous task also it is application dependent. The nonavailability of coarse-grained sense

inventory hampers the utilisation of the WSD system in a few NLP applications. The granularity is another bottleneck in the development of sense inventories and hence the WSD system.

(v) The availability of the large electronic corpora has encouraged the application of statistical techniques on the corpus for the WSD system. The statistical techniques lack deep linguistic processing or understanding of the natural language. The selection of feature vectors from the unstructured corpus data is another problem in statistical techniques. Currently, statistical techniques are incorporated in the WSD systems due to their automatic working features. However, the shortcomings of statistical techniques are hindering the development of the WSD systems.

(vi) The WSD systems are generally tested by comparing their results on a task against the results on the same task by human beings. Humans will not always agree on which word belongs to which sense. This problem is known as inter judge variance. To identify proper senses of words, one must know common sense facts. To inculcate common sense in a machine is a difficult task.

(vii) The identification of the standard WSD algorithms is an impossible task because the WSD task is dependent on the language, corpus availability and the use of external knowledge resources. It is impossible to manually create and annotate such vast datasets accurately because human annotators also tend to disagree.

(viii) Applications have resolved the ambiguity based on their needs. Nevertheless, the WSD is a holistic system which can miss application's subtle details. The knowledge of a domain of interest is required to prepare the WSD for a specific domain. The domain-oriented WSD such as biomedicine, computer science, tourism, and so on, need domain-specific text and domain tuning.

(ix) Sense disambiguation is highly dependent on language because each language has a different script, corpus and knowledge resource to identify sense. There is also much interest in cross-language sense disambiguation, where senses are looked up in an alternate language, using parallel corpora. This, linguists argue, simplifies the process of sense disambiguation because a multiple sense word can translate to a single sense in another language. The preparation of parallel corpus is a manual and complex process.

The WSD task is dependent on the natural language. The knowledge definition, selection and representation steps of the WSD are crucial for a natural language. In this research work, the

WSD for the Punjabi language has analysed, developed, and evaluated to address the listed challenges.

1.5 Need of WSD System for the Punjabi Language

India is a multilingual country having 22 national languages. The interlanguage processing tasks like machine translation, question answering, sentiment analysis, cross-lingual search, *etc*. are highly relevant problems in India. The emergence of the Indian languages on almost every web application such as Google, Amazon, Facebook and Twitter, *etc*. has necessitated examining the NLP tasks for the Indian languages. The Punjabi language is the official language of the Indian state of Punjab. It is classified as a member of the Indo-Aryan subgroup of the Indo-European family of languages. The Punjabi language is the world's 10th most widely spoken language (Kachru, et al., 2008). It is the third language of the Canadian parliament after the English and the French languages. The Punjabi language has also most widely spoken in Pakistani Punjab. The availability of enormous online text in the Punjabi language (Gurmukhi script) has motivated us to exploit the Punjabi WSD.

The researchers have developed the different Punjabi NLP applications for the Punjabi people of different world regions. The WSD is essential for the NLP tasks of Punjabi as well as other languages. In the Punjabi language, ambiguity exists at many levels like semantic, syntactic, internal word structure, and anaphoric. The core issue of the WSD for the Punjabi language is the absence of computational resources like a standard online dictionary, sense tagged or labelled datasets. To establish and evaluate the WSD task for the Punjabi language, unique characteristics of the Punjabi language need special consideration.

1.5.1 Characteristics of the Punjabi Language

The Punjabi language has been written in Shahmukhi, Gurmukhi and Devanagari scripts. The commonly used Punjabi scripts are Gurmukhi and Shahmukhi (Humayoun & Ranta, 2010). There are many differences in the Gurmukhi and Shahmukhi scripts. Both the scripts have different origins, a different set of alphabet, different directions of writing, and different kinds of writing system conventions. Gurmukhi and Devanagari scripts have originated from the Brahmi script. Shahmukhi has derived from Arabic script that is a local variant of Urdu script, and is used to record the Punjabi language. The medieval Punjabi literature has been written in the Shahmukhi script, while the modern literature of the Punjabi language is written in the Gurmukhi script.

In the whole thesis the Punjabi sentences are prefixed with P, its respective transliteration with TT, and English Translation with T. The software Gtrans performs the Gurmukhi to Roman script transliteration. It is accessible at the web address http://www.learnpunjabi.org /gtrans/index.asp.

The Punjabi language is written chiefly in east Punjab (India) using the Gurmukhi script (Virk, et al., 2011). The Gurmukhi script, like the roman script, is written left to right. The Gurmukhi script has forty-one letters, including thirty-eight consonants and three basic vowel sign bearers. It has ten clear vowel signs and three auxiliary signs. Table 1.2 shows the letters, consonants, vowel bearers, vowel signs, and auxiliary signs of the Gurmukhi script.

Table 1.2: Punjabi Letters in the Gurmukhi Script

Letters	ੳ, ਅ, ੲ, ਸ, ਹ, ਕ, ਖ, ਗ, ਘ, ਙ, ਚ, ਛ, ਜ, ਝ, ਞ, ਟ, ਠ, ਡ, ਢ, ਣ, ਤ, ਥ, ਦ, ਧ, ਨ, ਪ, ਫ, ਬ, ਭ, ਮ, ਯ, ਰ, ਲ, ਵ, ੜ, ਸ਼, ਖ਼, ਗ਼, ਜ਼, ਫ਼, ਲ਼
Consonants	ਸ, ਹ, ਕ, ਖ, ਗ, ਘ, ਙ, ਚ, ਛ, ਜ, ਝ, ਞ, ਟ, ਠ, ਡ, ਢ, ਣ, ਤ, ਥ, ਦ, ਧ, ਨ, ਪ, ਫ, ਬ, ਭ, ਮ, ਯ, ਰ, ਲ, ਵ, ੜ, ਸ਼, ਖ਼, ਗ਼, ਜ਼, ਫ਼, ਲ਼
Vowel bearers	ੳ, ਅ, ੲ
Vowel signs	ਅ, ਆ, ਇ, ਈ, ਉ, ਊ ਏ, ਐ, ਓ, ਔ
Auxiliary signs	ਂ, ਃ, ਁ
Zones of characters	

Gurmukhi characters have connected with the headline and can be partitioned into three zones. The upper zone is above the headline, the middle zone is below the headline, and the lower zone is below the middle zone as shown in Table 1.2. A brief description of the discrete characteristics of the Punjabi language is as follows.

(i) The Punjabi language is a phonetic and each of its characters represents a syllable. There exit aspirated, unaspirated, retroflex and dental phonetic features in the Punjabi language. Aspirated consonants are accompanied by an audible expulsion of breath, whereas non-aspirated consonants are produced with a minimal breath. Consonants with dots (ਸ਼, ਖ਼, ਗ਼, ਜ਼, ਫ਼, ਲ਼) consist of a dot in its foot and are used to represent the borrowed words from the other languages in the Punjabi language. The consonants are written in a conjunct form at a subscript to the main letter called subjoined consonants. There are three subjoined letters (ੁ, ੂ, ੃) which are written as a subscript of the letter (ਜ). The Punjabi language extensively

uses nasal characters. There are five consonants namely (ਙ, ਞ, ਣ, ਮ, ਨ) that are spoken using nasal cavity and two nasalization signs bindi (ਂ) and tippi (ਁ) that accompany consonants like (ਜਂ, ਜੰ). To allocate a double sound to the particular letter symbol adhak (ੱ) is assigned like (ਜੱ). The vowels in Punjabi are divided into independent and dependent forms. The dependent vowels are attached to a consonant, whereas independent vowels do not require any consonants.

(ii) In the Punjabi sentences, the subject occurs first, followed by the object and then the verb. The basic sentence structure of Punjabi is SOV (Subject-Object-Verb). The Punjabi language is relatively a free order language such that SOV, OSV, OVS, VOS and VSO can realize the same meaning. Out of them, SOV is frequently used by native Punjabi speakers. Table 1.3 shows Punjabi sentences with different word order but having the same meaning.

Table 1.3: Free word order examples

Sample Instance	**English Translation**
P: ਰਾਜ ਘੋੜਾ ਲੈ ਗਿਆ। TT: rāj ghōṛā lai giā	T: Raj took the horse.
P: ਘੋੜਾ ਰਾਜ ਲੈ ਗਿਆ। TT: ghōṛā rāj lai giā	T: The horse took Raj

(iii) The sentence structure of the Punjabi language can be classified into simple, complex, and compound. The simple sentence consists of an independent clause. The sentence with dependent and one or more independent clauses combined are called complex sentences. In compound sentences, two independent clauses are combined using conjunctions. The example of simple, complex, and compound Punjabi sentences are listed in Table 1.4.

Table 1.4: Examples of simple, complex and compound Punjabi sentences

Type	**Example Sentence**	**English Translation**
Simple	P: ਉਹ ਗੀਤ ਗਾਉਂਦਾ ਹੈ। TT: uh gīt gāundā hai	T: He sings a song
Complex	P: ਜੇ ਮੈਂ ਨਾ ਪਹੁੰਚਿਆ ਤਾਂ ਤੁਸੀ ਚਿੱਠੀ ਜਰੂਰ ਪਾ ਦੇਣਾ। TT: jē maiṃ nā pahuñciā tāṃ tusī ciṭṭhī jarūr pā dēṇā	T: If I do not reach you then must post the letter
Compound	P: ਮੈਂ ਸਾਇਕਲ ਤੇ ਜਾ ਰਿਹਾ ਸੀ ਅਤੇ ਰਸਤੇ ਵਿੱਚ ਮੀਂਹ ਪੈਣ ਲਗਾ। TT: maiṃ sāikal tē jā rihā sī atē rastē vicc mīṃh paiṇ lagā.	T: I was going on the bicycle and it started raining on the way

The subject, verb, and object components of sentences are classified into nouns, pronouns, verbs, adverbs, postpositions, *etc*.

Nouns of the Punjabi language are inflected for case (direct, oblique, locative, vocative, ablative) and number (singular or plural). Table 1.5 shows a Punjabi word and its respective case, number and gender. Punjabi noun words can be divide into extended and unexpended declensional types. The extended declensional nouns consist of masculine ending in unaccented '*aa*', whereas unexpended declensional nouns consist of feminises ending in '*ee*'. Though there are exceptions, most of the words in the Punjabi language can be distinguished as masculine and feminine using this rule. For Example, ਮੁੰਡਾ (TT: muṇḍā) means "*boy*" and ਕੁੜੀ (TT: kuṛī) means "*girl*". Similarly, ਬੱਚਾ (TT: baccā) is a "*male child*" and ਬੱਚੀ (TT: baccī) is a "*female child*".

Table 1.5: Punjabi noun word with its case, number and gender

Word	**Case**	**Number**	**Gender**
ਮੁੰਡਾ (TT: muṇḍā)	Direct	Singular	Masculine
ਮੁੰਡੇ (TT: muṇḍē)	Oblique and Direct	Singular	Masculine
ਮੁੰਡਿਆ (TT: muṇḍiā)	Vocative	Singular	Masculine
ਮੁੰਡਿਆਂ (TT: muṇḍiāṃ)	Oblique	Plural	Masculine
ਮੁੰਡਿਓ (TT: muṇḍiō)	Vocative	Plural	Masculine

(iv) There are six pronouns, namely personal, reflexive, demonstrative, indefinite, relative and interrogative available in the Punjabi language. Pronouns are also inflected for gender, number and case. The personal Punjabi pronouns such as 'ਮੈਂ' (TT: maiṃ, T: me), 'ਮੇਰਾ' (TT: mērā, T: mine), 'ਤੁਸੀਂ' (TT: tusīṃ, T: you), *etc*. have been used for the first and second persons, while for the third person demonstrative pronouns such as 'ਉਹ' (TT: uh, T: they), 'ਇਹ' (TT: ih, T: this) have used. The Punjabi demonstrative pronouns can be further categorized deictically as remote and near, respectively. Personal pronouns 'ਤੂੰ' (TT: tūṃ, T: you) and 'ਤੁਸੀਂ' (TT: tusīṃ, T) in the Punjabi language have the distinction in the number as well as nature (latter is a polite form of the earlier). 'ਕੋਈ' (TT: kōī), 'ਕੁਝ' (TT: kujh), 'ਹੋਰ' (TT: hōr), *etc*. are some of the indefinite pronouns used in the Punjabi language and 'ਸੋ' (TT: sō), 'ਜੋ' (TT: jō), 'ਜਿਹੜਾ' (TT: ijhaṛā), *etc*. are called the relative pronouns. 'ਕੌਣ' (TT:

kauṇ), ‘ਕਿਹੜਾ’ (TT: kihṛā), ‘ਕਿਹੜੀ’ (TT: ikhaṛī) are the interrogative pronouns used to ask questions in the Punjabi language. ‘ਕੌਣ’ (TT: kauṇ) pronoun is used for animate nouns whereas the other two are used for inanimate nouns. A few other interrogative pronouns of the Punjabi language are ‘ਕਿਸਦਾ’ (TT: iksadā), ‘ਕਿਵੇਂ’ (TT: ikvēṃ), *etc.*

(v) The Punjabi verb system is primarily structured around a combination of aspect and tense or mood. In a Punjabi sentence, verbs must agree with the subject or object of the sentence in terms of gender, number, and person. A Punjabi verb root is always in the second person singular form and the imperative mood. The Punjabi language has two aspects: the perfective and habitual, and possibly a third in the continuous, with each having over morphological correlations. Most of the verbs used in the Punjabi language are one word, but there are also two-word verbs called double and conjunct verbs. For example, ‘ਤੁਰ-ਤੁਰ ਕੇ’ (TT: tur-tur kē), ‘ਖਾਧਾ-ਪੀਤਾ’ (TT: khādhā-pītā), ‘ਖੇਡ-ਖੇਡ ਕੇ’ (TT: khēḍ-khēḍ kē) are a few double verbs used in the Punjabi language.

(vi) There are two auxiliary verb tenses, present and past, for the Punjabi language. The present tense auxiliary word of the Punjabi language is ‘ਹੈ’ (TT: hai) and the past tense auxiliary word is ‘ਸੀ’ (TT: see). It is also inflected for number and person. For instance, the inflected form of ‘ਹਾਂ’ (TT: hāṃ) and ‘ਸਾਂ’ (TT: sāṃ) for number are ‘ਹਨ’ (TT: hana) and ‘ਸਨ’ (TT: san). These forms can equally be used for both genders. There is no auxiliary verb for future tense in the Punjabi language.

(vii) Adjective of the Punjabi language can be categorized as inflected and uninflected. Inflected adjectives in the Punjabi language are inflected for the number, case and gender of nouns they qualify. For instance, in ‘ਛੋਟੀ ਬੱਚੀ’ (TT: chōṭī baccī, T: small girl) where ‘ਛੋਟੀ’ (TT: chōṭī) {Word Class: Inflected Adjective; Gender: Feminine; Number: Singular; Case: Direct;} is an adjective for word “*girl*” and ‘ਛੋਟਾ ਬੱਚਾ’ (TT: chōṭā baccā, T: small boy) where ‘ਛੋਟਾ’ (TT: chōṭā) {WordClass: Inflected Adjective; Gender: Masculine; Number: Singular; Case: Direct;} is an adjective for word “*boy*”. The uninflected adjectives are completely invariable and can end in either consonants or vowels. For Example, ‘ਉਦਾਸ’ (TT: udās, T: sad), ‘ਉਪਯੋਗੀ’ (TT: upyōgī, T: useful), ‘ਬਦਸ਼ਕਲ’ (TT: badshakal, T: ugly). All adjectives

can be used either attributively, predicatively, or substantively. Those used substantively are declined as nouns rather than adjectives.

(viii) Punjabi adverbs can be inflected or uninflected. Inflected adverbs inflect for gender, number, and case. For Example, 'ਤੇਜ਼' (TT: tēz, T: fast), 'ਪਿਛੇ' (TT: pichē, T: behind), 'ਅੱਗੇ' (TT: aggē, T: next), *etc*.

(ix) The Punjabi language has postpositions that link nouns, pronouns, and phrases to other parts of the sentence. Some postpositions of the Punjabi language are 'ਨੇ' (TT: nē, T: on), 'ਨੂੰ' (TT: nūṃ, T: to), 'ਉੱਤੇ' (TT: uttē, T: on), 'ਦਾ' (TT: dā, T: of), 'ਤੋਂ' (TT: tōṃ, T: from), 'ਵਿੱਚ' (TT: vicc, T: in/into), 'ਨਾਲ਼' (TT: nāḷ, T: with). Postpositions in the Punjabi language can also be inflected that changes form for gender, number and case. For example, 'ਉੱਪਰ' (TT: uppar, T: up), 'ਉੱਤੇ' (TT: uttē, T: on) postpositions become 'ਉੱਪਰੋਂ' (TT: upprōṃ, T: from above), 'ਉੱਤੋਂ' (TT: uttōṃ, T: on top) for the ablative case as shown in Table 1.6.

Table 1.6: Example of the ablative case of Punjabi preposition

Example Sentence	**English Translation**
P: ਉਸ ਨੇ ਮੇਜ਼ ਉੱਪਰੋਂ ਆਪਣੀ ਕਿਤਾਬ ਚੁੱਕ ਲਈ। TT: us nē mēja੍ upprōṃ āpṇī kitāb cukk laī	T: He picked up his book from the table.
P: ਬੱਚੇ ਨੇ ਮੰਜੇ ਉੱਤੋਂ ਛਾਲ਼ ਮਾਰੀ। TT: baccē nē mañjē uttōṃ chāla੍ mārī.	T: The child jumped from the bed

(x) The Punjabi conjunctions combine two clauses in a sentence or two words in a clause. These words are uninflected and are categorized based on the type – co-ordinate and subordinate. Co-ordinate conjunctions, like 'ਅਤੇ' (TT: atē, T: and) join two independent clauses in a compound sentence. These can also act as connectives for various phrases and words of word classes, like noun, pronoun, verb, adjective, and adverb. Subordinate conjunctions, like 'ਇਸ ਲਈ' (TT: is laī, T: because), join an independent and a dependent clause in a complex sentence.

(xi) The particles of the Punjabi language like honorific 'ਜੀ' (TT: jī, T: yes), negative 'ਨਹੀਂ' (TT: nahīṃ, T: No), 'ਨਾ' (TT: nā, T: No), and 'ਹੀ' (TT: hī, T: already), 'ਤਾਂ' (TT: tāṃ, T: so), 'ਵੀ' (TT: vī, T: also) are uninflected.

(xii) The Punjabi language has a severe spelling variation problem. For instance, two different spelling 'ਵਿਚ' (TT: vic), 'ਵਿੱਚ' (TT: vicc) are commonly used in the Punjabi language. The nonavailability of the Punjabi spelling resources has aggravated spelling issues in the Punjabi language corpus. The spelling issues have increased the pre-processing task of cleaning the corpus data with standard spelling.

(xiii) The Punjabi language has included numerous words from other languages, namely the words from the Hindi, Urdu, Persian and English languages. The Punjabi language users frequently use the English language words in the context. This makes the analysis of the Punjabi language context a complex task. The English words "*percent*" and "*sir*" are used in the respective two sentences, as shown in Table 1.7.

Table 1.7: Examples of English words used in Punjabi sentences

Example Instance	**English Translation**
P: ਸੈਂਟ ਪ੍ਰਸੈਂਟ ਅੱਜ ਮੀਂਹ ਪਉਗਾ। TT: saiṇṭ prasaiṇṭ ajj mīṃh paugā	T: It will rain today 100 percent.
P: ਸਰ ਜੀ ਮੇਰੀ ਬੇਨਤੀ ਤਾਂ ਸੁਣੋ। TT: sar jī mērī bēntī tāṃ suṇō	T: Sir, please listen to my request.

The Hindi and Punjabi languages have high redundancy values as compared to other languages. Thus data comparison is more expensive for the Hindi and Punjabi languages than the other South Indian languages (Bharati, et al., 2000). There exist critical differences between the Indo Aryan Punjabi language and the English language. The Punjabi text needs special attention before performing WSD classification by the same classifiers that perform English WSD. The next subsection explains the difference between English and Punjabi text.

1.5.2 Diversion of Punjabi over the English Language

There are many prominent differences between the Punjabi and English languages and some of the key points observed are listed below.

(i) The commonly used sentence structure of the Punjabi language is SOV (Subject-Object-Verb) as compared to the English language SVO (Subject-Verb-Object) sentence structure.

(ii) The Punjabi language is relatively a free word order language while the English language is the fixed order language.

(iii) The Punjabi language is a null-subject language where the subject is not required for the completion of a sentence, while the English language usually uses "*It*" and "*There*" to complete the sentences. Table 1.8 shows the two example sentences of the Punjabi language having null-subject.

Table 1.8: Null subject in Punjabi sentences

Sample Instance	**English Translation**
P: ਬਾਹਰ ਮੀਂਹ ਪੈ ਰਿਹਾ ਹੈ। TT: bāhar mīṃh pai rihā hai	It is raining outside.
P: ਲੜਾਈ ਸ਼ੁਰੂ ਹੋ ਸਕਦੀ ਹੈ। TT: laṛāī shurū hō sakadī hai	There could be a war.

(iv) In the Punjabi language, two gender forms, masculine and feminine, for the same noun word are frequent. The two example sentences in Table 1.9 show that the same word ਅਖਬਾਰ (TT: akhbār, T: 'newspaper') of the Punjabi language has been treated as masculine and feminine.

Table 1.9: Gender overlap examples of Punjabi sentences

Sample Instance	**English Translation**	**Gender**
P: ਰਾਮ ਨੇ **ਅਖਬਾਰ** ਪੜ੍ਹ ਲਿਆ ਹੈ। TT: rām nē akhbār paṛh liā hai	Ram has read the newspaper.	masculine
P: ਰਾਮ ਨੇ **ਅਖਬਾਰ** ਪੜ੍ਹ ਲਈ ਹੈ। T: rām nē akhbār paṛh laī hai	Ram has read the newspaper.	feminine

(v) The Punjabi language is highly inflectional as compared to the English language. It has rich morphological variations for the words having PoS classes, namely verbs, adjectives, and nouns.

(vi) Most of the noun words in the Punjabi language can be distinguished as masculine and feminine using declensional subtypes, but no such distinction exists in the English Language.

(vii) The English language has no inflections for adjectives corresponding to the nouns they qualify, while the Punjabi language has inflected adjectives that are marked, through termination, for the gender, number and case of the nouns they qualify.

(viii) The Punjabi language has postpositions rather than prepositions of the English language. The prepositions are used at the beginning of the English phrase, while postpositions are used at the end of the Punjabi phrase. The postpositions of the Punjabi language are also inflectional.

(ix) English translation of reflexive pronoun is different for demonstrative and personal pronouns. For instance, in Table 1.10, sentences 'ਆਪਣਾ' (TT: āpaṇā) is used in Punjabi whereas in English, "*my*" is used for "*I*" and "*his*" for "*he*".

Table 1.10: Reflexive pronoun examples

Sample Instance	**English Translation**
P: ਮੈ ਆਪਣਾ ਕੰਮ ਕਰ ਰਿਹਾ ਸੀ। TT: maiṃ āpaṇā kamm kar rihā sī	I was doing my work
P: ਉਹ ਆਪਣਾ ਕੰਮ ਕਰ ਰਿਹਾ ਸੀ। TT: uh āpaṇā kamm kar rihā sī	He was doing his work

(x) Punjabi verb participle forms, inflecting for gender and number by way of vowel termination. For example, Punjabi verbs like ਗਈ (gaī) {WordClass: Main Verb; Gender: Feminine; Number: Singular;}, ਗਏ (gaē) {WordClass: Main Verb; Gender: Masculine; Number: Plural;} and ਗਿਆ (giā) {WordClass: Main Verb; Gender: Masculine; Number: Singular} have only one equivalent "*gone*" in English.

(xi) Punjabi language main verbs have 48 distinct forms on average, unlike the English verbs, which usually have only five different forms.

(xii) The nouns of the Punjabi language are inflected for case and number. For instance, there are five inflectional forms of word ਮੁੰਡਾ (TT: muṇḍā), ਮੁੰਡੇ (TT: muṇḍē), ਮੁੰਡਿਆ (TT: muṇḍiā), ਮੁੰਡਿਆਂ (TT: muṇḍiāṃ), ਮੁੰਡਿਓ (TT: muṇḍiō), whereas it has only one English equivalent "*boy*".

(xiii) Adjectives in the Punjabi language are inflected for number, case and gender of nouns they qualify. In the case of the English adjectives, they are not inflected based on the number form of the nouns.

(xiv) The Punjabi language has two-word verbs called double and conjunct verbs, while the English language does not have such double and conjunct verbs.

(xv) The particles of the Punjabi language like ਜੀ (TT: jī), and ਹੀ (TT: hī) do not have any equivalent in the English language.

(xvi) Besides the differences from the English language, there are some common issues like ambiguous newspaper headlines and distinctions of the same verb and nouns, as shown in the example instance of Table 1.11.

Table 1.11: Newspaper headline with the same noun and verb examples

Issues	Sample Instance	English Translation
Ambiguous newspaper headline.	P: ਆਮ ਆਦਮੀ ਦੀ ਹੈ ਆਮ ਆਦਮੀ ਪਾਰਟੀ। T: ām ādmī dī hai ām ādmī pāraṭī	Aam Aadmi Party is people's party.
Same verb and noun.	P: ਪੂਜਾ ਨੇ ਪੂਜਾ ਦੀ ਥਾਲੀ ਨਾਲ ਪੂਜਾ ਕੀਤੀ। T: pūjā nē pūjā dī thālī nāl pūjā kītī	Pooja has worshiped using a plate.

These different structural and grammatical challenges of the Punjabi language make the WSD task harder. The WSD system needs a robust morph analyser to understand these rich variations in attributes of the Punjabi context words and mine the ambiguous word's proper sense. Notably good pre-processor for the Punjabi language is required, to take care of spelling and multi-lingual issues.

1.6 Gap Analysis

The Punjabi language has unique characteristics that need special attention to determine the sense of the ambiguous words using the WSD. The explicit Punjabi WSD needs accessibility to structure and unstructured knowledge resources of the Punjabi language. To prepare the knowledge resources for the Indian Languages, the Indian Institute of Technology Bombay had created the Hindi WordNet as a first step. In order to expand the WordNet creation activity to other Indian languages, the IndoWordNet project has been initiated. In this project, Hindi language WordNet is extended to all other major Indian languages (Bhattacharya, 2010). The creation of WordNet for the Punjabi language is one of such efforts (Narang, et al., 2013). The availability of Punjabi WordNet, a lexical resource, has motivated us to implement an explicit Punjabi WSD system. However, to implement Punjabi WSD, the research gaps have been identified and are presented as follows.

(i) The WSD is the core part of the language computational tasks. Indian languages have unique characteristics that need to be addressed to implement the WSD system or other NLP tasks.

(ii) An exhaustive survey on the WSD for Indian languages is not available. Hence there is a need to perform a detailed analysis of the WSD for Indian languages to extract the knowledge about the existing corpus (raw and annotated), knowledge bases and the different techniques of the WSD.

(iii) There exists little research work for the supervised WSD of the Punjabi language. This is due to the non-availability of the labelled data and the limited availability of lexical resources for the Punjabi language to perform the WSD of Punjabi text. There is a necessity to create sense labelled datasets for the Punjabi language.

(iv) A variety of classification techniques have been explored for performing the WSD for different languages. There are no standard classification algorithms that have performed well for the WSD task. In order to explore the behaviour of different classification algorithms on the Punjabi language WSD, they need to experiment on the Punjabi text.

(v) The procedure of selection and application of lexical, syntactic and semantic features from labelled datasets for the supervised Punjabi WSD has to be explored. The impact of lexical, syntactic and semantic features for the Punjabi WSD has to be studied.

(vi) There is a need to analyse the impact of prediction-based word embedding features for the Punjabi WSD.

(vii) The application of the deep learning approaches has improved the coverage and performance of the WSD for different languages. There is a need to study the impact of deep learning approaches for the Punjabi WSD classification task.

(viii) There is no implicit or explicit WSD module available for the Punjabi language. This module can be used in the various natural language processing applications of the Punjabi language like information retrieval, information/text extraction, content analysis, word processing, lexicography, semantic web and machine translation.

(ix) There exist no explicit Punjabi WSD system on the web that classifies the sense of the ambiguous Punjabi words.

(x) The evaluation of WSD approaches needs standard data. No standard labelled dataset is available for the Punjabi language.

The following research objectives have been formulated after investigating the above-mentioned research gaps.

1.7 Research Objectives

In order to fill the gaps, an explicit Punjabi language WSD has developed as it enables most of the NLP tasks to perform optimally. The core objective of this research proposal is to create a word sense disambiguation module for the Punjabi language. The following objectives have been proposed to achieve the Punjabi WSD task.

(i) To analyse various word sense disambiguation classification techniques like supervised, semi-supervised, unsupervised and knowledge based.

(ii) Compare different classification techniques of word sense disambiguation for the Punjabi language.

(iii) Create a manual sense tagged corpus for the Punjabi language with the use of the Punjabi WordNet, which can act as a trained set for training of the proposed system.

(iv) To implement the word sense disambiguation module for the Punjabi language.

(v) Evaluate proposed word sense disambiguation system for the Punjabi language on standard evaluation techniques like Senseval/Semeval.

1.8 Research Methodology

The objectives of this research work have accomplished by applying the following research methodology.

To achieve the first objective

- Several research studies have examined to determine the classification approaches explored for the WSD by the researchers.
- The different supervised, semi-supervised, unsupervised and knowledge-based classification techniques of the existing techniques for the WSD have been analysed.
- The different sense inventories, knowledge resources and the labelled datasets for the WSD of different languages have been reviewed.
- The use of various classification approaches by the Indian languages for the sense disambiguation has been comprehensively reviewed
- The application of different annotated lexical and linguistic resources available for different Indian languages for the WSD has been identified

To achieve the second objective

- A comparative analysis of different Punjabi WSD approaches has performed based on the different parameters like usage of knowledge base, ambiguous words used, different features and the performance metrics.
- An evaluation of the different WSD methodology (knowledge base) for the Punjabi language has been performed.
- The impact of different supervised classifiers such as statistical-based, kernel-based, probabilistic, logic-based, ensemble-based and tree-based has been analysed on the Punjabi dataset.
- A comparative analysis has been performed between supervised and neural network deep learning classifiers.

To achieve the third objective

- The ambiguous Punjabi language words have been selected from the lexical database WordNet to prepare the labelled Punjabi dataset.
- The 150 ambiguous Punjabi noun words are selected from Punjabi WordNet. The words having senses greater than or equal to 2 have been selected from the Punjabi WordNet.
- Different supervised, sense similarity, overlapping different inventories, semi-supervised, translation similarities and sense merging approaches have been analysed.
- A set of semantic relational rules have applied individually on the fine senses of 150 ambiguous Punjabi noun words to generate their respective coarse senses.
- The labelled dataset of 150 ambiguous Punjabi noun words is prepared manually by carefully collecting the instances for the individual coarse sense of the independent word from the online and offline resources.
- The kappa statistic has applied to the sense tagged Punjabi dataset to measure the level of agreement among the native Punjabi speakers who labelled it.

To achieve the fourth objective

- The explicit targeted Punjabi WSD system based on supervised classifiers has been proposed and implemented.
- The sense tagged dataset is pre-processed. The noise such as stop words, punctuations, numeric characters, *etc*. is removed from the dataset. The dataset is further normalized by removing multiple spellings and expanding acronyms.
- The lexical, syntactic and semantic contextual representations of the dataset have been devised, also called features, to experiment on machine learning and deep learning classifiers.
- The lexical features such as the bag of words, collocations, co-occurrence, unigram bigram and trigram are derived from the labelled Punjabi datasets. The syntactic feature like part of speech is selected from the part of speech labelled Punjabi datasets. The word embedding features are prepared from the unlabelled Punjabi datasets (like Wikipedia, online books, *etc.*).
- The supervised classifiers have been trained on the feature vectors and are further validated in the training process using the ten-fold cross-validation approach.
- The web interface of this architecture has been implemented that applies the trained classifiers to distinguish different senses of the ambiguous word.

To achieve the fifth objective

- The accuracy is the measure of evaluation of the WSD systems for different languages that participate in the standard Senseval/Semeval competitions. The proposed WSD system has been evaluated based on accuracy parameters.
- Punjabi language that has no standard dataset available at Senseval/Semeval competitions. The different test datasets of individual words have been used manually prepared and are extensively applied to evaluate the Punjabi WSD system.
- The Punjabi WSD system has been compared with existing WSD systems of different Indian languages.

The detailed research contributions of this work after following the research methodology to accomplish the research objective are discussed in the following section.

1.9 Research Contributions

The significant contribution of this thesis is the development of an explicit targeted WSD system for the Punjabi language. The detailed description of different contributions is as follows.

- An exhaustive literature review has conducted to determine the contemporized sense inventories and classification approaches explored by researchers for the WSD of different Indian languages. The comparison between different supervised, unsupervised and knowledge based classification has been performed. It has been observed that the supervised WSD methods have outperformed all other approaches of targeted WSD.
- The literature review for the WSD for Indian languages has been performed and it has been observed that there is minimal work has been done for the Punjabi language WSD.
- The availability of the Punjabi WordNet that describes the synonym words and their relationships has encouraged us to use it as the sense inventory for this research. The most ambiguous noun words have been selected from the Punjabi WordNet to prepare the sense tagged dataset for supervised WSD.
- The algorithm has been devised to transform the fine senses of the Punjabi WordNet to coarse senses of Punjabi noun words. The sense annotated dataset for 150 ambiguous words has been prepared manually by multiple Punjabi native speakers. The kappa statics validate the sense tagged dataset of the Punjabi language.
- The pre-processing algorithm has been developed to clean the noise from the annotated trained dataset. A dictionary of acronym and stop word lists have been prepared for the Punjabi language required during the pre-processing activity.

- The different lexical (topical and local), syntactic and semantic feature sets have been explored and applied to machine learning and deep machine learning classifiers.
- The lexical unigram, bigrams, trigrams, collocations, co-occurrence and bag of word features are selected from the labelled Punjabi dataset. The part of speech syntactic collocation feature is selected from the part of speech labelled Punjabi datasets. The semantic term frequency-inverse document frequency and co-occurrence matrix have been selected from labelled datasets. The word embedding semantic features are devised from the unlabelled Punjabi corpus like the Punjabi Wikipedia.
- A supervised Naïve Bayes classifier has been developed specifically for the WSD of Punjabi language and experimented individually on 150 ambiguous Punjabi noun words.
- The performance of the Naïve Bayes, Decision Tree, Random Forest, Decision List, Support Vector Machine, k Neareat Neighbour supervised classifiers used for Punjabi WSD has been compared based on different feature sets, the number of senses of words and the instance distribution among the different senses of words.
- The ensemble machine learning classifiers have experimented on the feature sets of Punjabi noun words. A comparison between ensemble and base machine learning classifiers has been performed.
- The deep learning classifiers have experimented on the Punjabi dataset. The neural network based embedding vectors Word2Vec, Global Vector (GloVe) and FastText have been prepared from the Punjabi Wikipedia. These embedding vectors have been used by both machine learning and deep learning classifiers for the WSD of the Punjabi language.
- A web-based explicit targeted Punjabi WSD has been designed and developed. The input to this web-based application is the Punjabi sentence. The ambiguous target word is identified from this input instance. The context of the target word is pre-processed and is represented in different lexical and semantic features. Based on the selected feature, this system applies trained models corresponding to the selected feature to predict the meaning of the target word ambiguous word.

Finally, these research contributions are incorporated chapter wise into the thesis.

1.10 Thesis Organisation

This thesis has been structured into eight chapters. After introducing the WSD in this chapter, a brief overview of the rest of the seven chapters is as follows.

Chapter 2 presents the detailed literature review of the WSD system based on different approaches and resources utilized by the researchers. An extensive literature evaluation has conducted by following a systematic approach. In this systematic approach, firstly, the related research studies have searched from the electronic repositories of top journals as well as the conferences related to the WSD. Then the inclusion and exclusion conditions have followed in selecting the most relevant research studies. Finally, the selected research studies have examined based on the formulated research questions. The evolution of the WSD for Indian languages and the timeline of research studies of the WSD for Indian languages have included in this chapter. The research conducted by the researchers on different WSD parameters such as Indian languages, classification techniques, corpus used and domains has covered in this chapter. The details of the structured and unstructured knowledge resources of the Indian Languages explored by the researchers for the WSD have presented in this chapter. The different context processing resources for Indian languages have listed in this chapter. The different WSD competition tasks held in standard Senseval/Semeval workshops are specified in this chapter. This chapter states the detailed findings of all the research questions formulated to conduct this systematic review.

Chapter 3 explains the process of creation of sense tagged corpus for the ambiguous Punjabi noun words. The sense tagged dataset is prepared by selecting the 150 target ambiguous words from the Punjabi WordNet. The structure of Punjabi WordNet is discussed in this chapter. The detailed process of conversion of fine senses to coarse senses has also been discussed. The sense labelled dataset for the individual target ambiguous words has been prepared by collecting instances for the individual senses of the ambiguous word. The annotated dataset has been validated using the kappa statistic measure. The details of the sense tagged dataset like the number of fine senses, coarse senses and number of instances per sense are presented in this chapter.

Chapter 4 describes the architecture of the explicit Punjabi WSD that disambiguates the targeted ambiguous words from the Punjabi text. A brief description of the different phases like pre-processing of text, parsing of text, feature selection methods, cross-validation training of the machine learning classifiers have been presented in this chapter. The collected sense tagged data has been pre-processed by removing the font variations, standardizing multiple spelling of words and abbreviation completion. The pre-processed sense tagged dataset has parsed to extract root words and PoS tags of the dataset. The implementation of the WSD for Punjabi language using statistical, logical, kernel-based, tree-based and ensemble classifiers

has been described. The deep learning classifiers such as artificial neural networks and long short term memory approaches for the Punjabi WSD have also been discussed in this chapter.

Chapter 5 presents the importance and the different methods of the context representation of the sense tagged dataset. This chapter explains the need for the different features for the WSD system. The feature selection phase of the Punjabi WSD has been presented in this chapter. The individual algorithms for the selection of topical (bag of word and co-occurrence), local (collocation and n-gram), syntactic (part of speech tags) and semantic (count and prediction-based) features are described in this chapter. The topical, local features have been selected from the sense labelled 150 ambiguous Punjabi words. The syntactic features are derived from the PoS tagged dataset of the ambiguous words. The count based term frequency and inverse document frequency feature has been prepared from the sense labelled dataset. The predictions based word embedding features have generated using the unlabelled Punjabi corpus (Punjabi Wikipedia). This chapter also describes the Word2Vec, GloVe and FastText neural network models used to select the different word embedding features of the Punjabi language. The word embedding features of the Punjabi language have been evaluated by performing the similarity and analogy completion tasks.

Chapter 6 explains the classification phase of the Punjabi WSD architecture. The string to vector algorithms have been devised to transform the selected string features such as topical, local and syntactic into the numeric vectors. The training of the machine learning classifiers using the topical, local and syntactic features have been discussed in this chapter. The word embedding features have been processed using concatenation, average, fractional and exponential techniques to train the machine learning classifiers. The deep learning classifiers have been trained using the word embedding features. The configuration parameters of the deep learning classifiers, namely activation function, learning rate, optimization algorithm, layer structure, batch size, epoch hyperparameters, are presented in this chapter. The design, development and implementation of a web-based WSD system that uses different supervised classifiers and features are presented in this chapter. This chapter explains the integration of machine learning and deep learning classifiers with the web. The data structures, file structures and output details of the Punjabi WSD systems are presented in this chapter.

Chapter 7 includes a discussion about the results of the Punjabi WSD system. The performance of the WSD system for the Punjabi language has been estimated by calculating the accuracy metric for the individual ambiguous word. This chapter shows the average

accuracy of the nine machine learning and two deep learning classification techniques based on different feature sets. The error analysis of the Punjabi WSD system has been discussed in this chapter. The supervised Punjabi WSD system has been compared with the existing supervised WSD systems of different Indian languages.

Chapter 8 concludes and presents the future implications of the research work presented in this thesis. It has been concluded that the results obtained from the Punjabi WSD system are very encouraging. Therefore, the Punjabi WSD system can be helpful for other natural language processing tasks which in turn will benefit society. In future, to increase the training dataset of the ambiguous words, the predicted dataset would be included. This targeted WSD system can be scaled further by adding more ambiguous Punjabi noun, verb, adverb and adjective words. The performance of the ensemble classifiers can be improved by including distributional architecture for the training of the individual base classifiers. To integrate this explicit Punjabi WSD system with the other NLP tasks, API's would be prepared. In the future, this integration would enhance the performance of other challenges such as machine translation, sarcasm detection, question answering tasks of the Punjabi language.

Chapter 2 : Literature Survey

This chapter provides a systematic review of Word Sense Disambiguation (WSD) for the Indian languages. The evolution of WSD for Indian languages and its present status has been presented in this chapter. A review methodology has been followed, and different research questions have been framed to perform the detailed and comprehensive survey of WSD for Indian languages. The research studies of WSD for the Indian languages have been searched from renowned electronic databases and the topmost conferences. The inclusion and exclusion conditions have delineated and followed in selecting the most relevant research studies of the WSD for Indian languages.

The details of the Indian languages and their families have discussed in this chapter. The evolution of WSD for the Indian languages and the timeline of research studies of WSD for Indian languages belonging to different language families have included in this chapter. From the shortlisted studies of the WSD for Indian languages, the status of research work has been determined by tabulating the different approaches, resources, datasets and evaluation measures used to perform WSD for all Indian languages. The list of structured (WordNet) and unstructured (labelled and unlabelled corpus) knowledge resources applied for WSD of the Indian languages in different studies have been presented. The context processing resources available and explored by researchers for the Indian languages have also presented in this chapter. The evaluation metrics explored by the researcher for WSD research of the Indian languages have been presented. This chapter has also presented the different WSD competition tasks held in standard Senseval/Semeval workshops that have executed the comparative evaluation of the WSD systems. Finally, the outcomes of this comprehensive survey have mapped with the framed research questions.

2.1 Evolution of the WSD

The automatic sense disambiguation has experimented with as a fundamental task in Machine Translation (MT) since the year 1940 (Hutchins, 1999). These earlier WSD methods have involved the processing of frequency of words but they have limited scope due to the limited size and domain of the micro-glossaries. Subsequently, the researchers have revealed beside the statistical information about words and their sense, the context words, semantic information of context and the detailed knowledge base of the text can improve the sense disambiguation task. In 1960 the performance of MT has stagnated because of the processing elements of WSD, such as the context in which a target word occurs, frequency of words and senses, knowledge

resources. It has required huge computational resources. In the 1970s, Artificial Intelligent (AI) techniques had incorporated to solve the Natural Language Processing (NLP) tasks. The automated WSD problem has been categorized into an AI problem due to its requirement of complete language acquaintance. The knowledge bases to store the complete language information have been manually crafted and have been used by automated WSD. But, these resources still lack scale and their usage in the WSD system had generated limited results. In the mid-1980s, the advancement in computational resources, automatic knowledge extraction methods and the availability of labelled corpora have encouraged the usage of machine learning techniques to solve the WSD problem. The WordNet, statistical revolution in NLP and Senseval workshops were major developments in the field of WSD that happened in 1990. The detailed history of WSD was discussed by (Ide & Véronis, 1998).

Automatic sense tagging and automatic sense disambiguation have been successfully implemented and evaluated in several languages since 1990. The availability of exponential online data, minimization of the use of sense tagged corpus, and the use of hybrid WSD systems reinvented the WSD field in 2000. However, the focused language was the English language, but in the last decade, this computational linguistic task has been explored and evolved in the context of the different languages of the World. Currently, the application of powerful machine-learning methods due to the availability of thousands of relatively simple processing cores on a single chip, the availability of an enormous amount of web data and the emergence of neural networks have significantly improved the multi-class and high dimension WSD task.

India is a multi-lingual country where 121 mother tongues exist, out of which 22 languages are the official languages (Government, 2011). India has the second-largest digital population and has the largest Facebook 290 million users (Clement, 2020). Therefore, the development of the NLP tasks is essential for effective communication among the diverse population of the Indian sub-continent, and the WSD is an essential component for these NLP tasks. The following section describes a systematic approach followed to examin the literature review of the WSD field applied for the Indian languages.

2.2 Review Methodology Followed

The review methodology follows a systematic approach, to analyse the WSD methods, knowledge resources, corpus and classification techniques used for different Indian languages. This systematic approach, have searched the related research studies from the electronic repositories of top journals as well as the conferences related to the WSD field. Then the

inclusion and exclusion conditions have followed in selecting the most relevant research studies. Finally, a detailed structured analysis has been performed to review the WSD and its role for Indian languages taking into account the framed research questions listed in Table 2.1.

Table 2.1:Research questions for systematic literature review

Research question	**Motivation**
RQ1: What is the timeline of research for the WSD for Indian languages and which database resources are used for research publication of Indian languages?	Identify the year wise status of the WSD for Indian languages. Identify the sources and the type of publication of the research studies.
RQ2: For which Indian languages WSD have been applied?	Categorizes the different Indian languages into different language families and determine the WSD applied for these different Indian languages.
RQ3: Which important lexical resources exist, developed or implemented for the Indian languages WSD ?	Identify the knowledge resources used for the WSD of Indian languages. The availability of lexical, linguistic and annotated datasets suggests the ease of development of WSD.
RQ4: What are the context processing tools available for Indian languages ?	The context processing tools for Indian languages are essential for the WSD as they extract the optimal feature text and handle Indian language specific attributes. Identify the availability of the different context processing tools for Indian languages.
RQ5: What are the different approaches of WSD?	Search the different approaches and domains of the WSD research. Identify the most commonly used WSD approaches.
RQ6: What are the different evaluation methods used to evaluate the WSD?	Find standard metric used to evaluate the WSD. List the standard evaluation methods used in SENSEVAL/SEMEVAL workshops of WSD field.
RQ7: Is an online/offline WSD system for Indian languages available?	Explore the different WSD systems available for the Indian languages.
RQ8: What are the future directions identified from the literature review?	Examine the literature review to determine the future direction in the field of WSD.

2.2.1 Search Resources and Criteria

The different information resources have been examined to extract the information about the research done in the WSD field. Based on the research questions listed in Table 2.1, the search criteria of the WSD and its role for Indian languages has been carried out via renowned electronic databases as per Kitchenham's guidelines (Kitchenham & Stuart, 2007). The following resources have been explored to collect information regarding WSD.

- ACM Digital Library (www.dl.acm.org)
- Springer (www.springer.com)
- IEEE Xplore (www.ieeexplore.ieee.org)
- Academia (www.academia.edu)
- CiteSeerX (www.citeseerx.ist.psu.edu)
- ResearchGate (www.researchgate.net)
- Association for the Advancement of Artificial Intelligence (www.aaai.org)
- Association for Computational Linguistics (www.aclweb.org)
- Google Scholar (www.scholar.google.co.in)

To find the relevant search studies from the e-databases, a keyword-based search has conducted, in which the vital keyword search strings have been prepared from the research questions enumerated in Table 2.1. The keywords used for the extraction of the useful WSD research studies has been registered in Table 2.2.

Table 2.2: Search strings used to find the studies from e-resources

Research Question	**Search String**
RQ1:	Initiation\Beginning of research in the WSD [for\in\of][Indian languages]
RQ2:	WSD [for\in\of] [Indian languages]
RQ3:	Knowledge based, unsupervised and supervised sense disambiguation [for\in\of] [Indian languages]
RQ4:	[Computational resources] for WSD [for\in\of] [Indian languages]
RQ5:	WSD tools [for\of] [Indian languages]

2.2.2 Inclusion Exclusion Process

The keywords "Word sense disambiguation" and "Sense disambiguation" have returned a large number of results as this field is explored for different languages in different domains. The most commonly returned research studies belong to the English language. To incorporate the research studies of the WSD for Indian languages, the name of the individual Indian language has been included in the search string. For instance, the keyword "Indian languages" from the search string "WSD [in, of, for] [Indian languages]" has been replaced with 22 official Indian languages one by one.

The search has performed in the abstract and the title of the research studies using the search string "WSD [in, of, for] [Indian languages]" to include the pertinent research studies of Indian languages. For example, some of the research studies have substrings like "in Punjabi" or "for Punjabi" or "of Punjabi" in their title. Therefore, to take into consideration of all the research

studies of the field, a search has been performed to include all these scenarios. The search string for each research question has mentioned in Table 2.2. These search strings have been taken into account to extract the research material from different research databases. The e-databases have returned a large number of titles. The following constraints have been defined to exclude the irrelevant WSD research studies.

- The timeline for the publication of the papers of WSD has been set from 2009 to 2020 as the major work on the WSD for Indian languages commenced in 2010.
- The research studies which have any one of the keywords mentioned in Table 2.2 in their titles have included. Otherwise, the study has not been considered.
- The redundant papers returned by the different e-resources have been carefully removed.
- The titles that have matched with the keywords stated in Table 2.2 have been selected. Further, the abstracts of the papers have been examined. The research studies that have relevance in the field for the WSD have been considered.
- Finally, the shortlisted papers based on the abstract have been studied and included only if their similarity towards this research topic exists.

The framed constraints have been carefully applied to the individual research study of the WSD returned by different e-resources. The research studies from many conferences, journals, workshops, master and PhD thesis have returned by search strings. Also, to complete the search for the WSD for the Indian languages some conferences and journals related to NLP and linguistics have also explored. Table 2.3 gives the details of publication types (Conferences (C), Journal (J) and Others (O)) of the research studies returned by e-databases.

Table 2.3: Publication types of research studies

Source	**Publication Type**	**Number of Results**
Google scholar	C/J/O	743
IEEE explore	C	134
ACM Digital library/ACL	C/J	119
COLING	C	103
Science Direct	J	65
Academia	C	15
CiteSeerX	C	12
ResearchGate	C	8

The staged process of application of exclusion constraints on the research studies has depicted in Figure 2.1. In the first step, the studies returned by web resources using the keyword search strings are 1200. Out of 1200, studies 867 have been shortlisted based on the title of the studies,

which have been reduced to 354 by studying the abstract of the studies. Finally, 110 research studies have been included in this review by studying the full text of 354 papers.

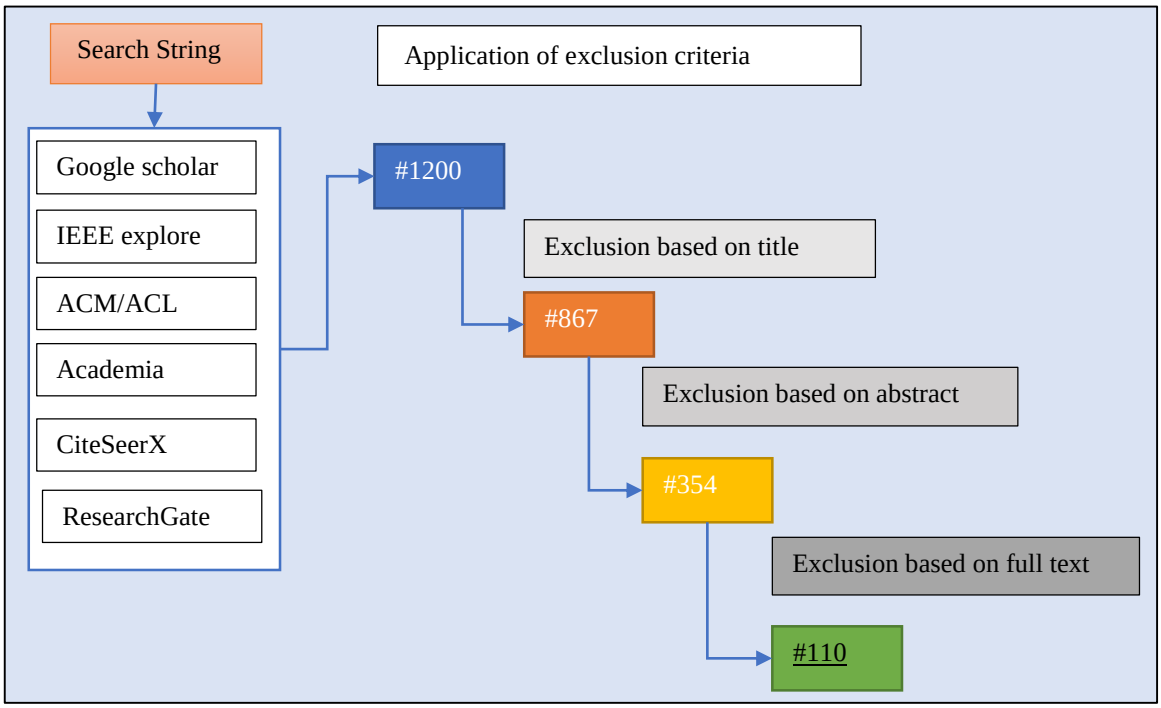


Figure 2.1: Review procedure followed

The following sections and sub-sections provide a detailed analysis of shortlisted research studies of the WSD for Indian languages. The analysis of the selected research studies has mapped with the research questions listed in Table 2.1.

2.3 Indian languages and their Families

India is a multilingual country. As per the 2011 census, Indians have 121 different languages, including 22 official scheduled languages (Government, 2011). Of the total population of India, 96.71% have one of the scheduled languages as their mother tongue, the remaining 3.29% have accounted for by other languages (Government, 2011). Official Indian languages have been broadly classified into Indo-Aryan, Dravidian, Sino Tibetan and Austro-Asiatic classes, as shown in Figure 2.2. These languages have been further categorised based on the regions in which they have been used.

(i) The Indo-Aryan family is the most spoken language family in India. Nearly 74% of the Indian population uses langauges of the Indo-Aryan family. Hindi, Bengali, Punjabi, Odia (Oriya), Nepali, Konkani, Marathi, Gujarati, Sindhi, Assamese, Dogri, Urdu, Kashmiri and Sanskrit are the major langauges of the Indo-Aryan family.

(ii) The Dravidian language family is the second largest language family in Indian language families. The main languages of the Dravidian language family are Telugu, Tamil,

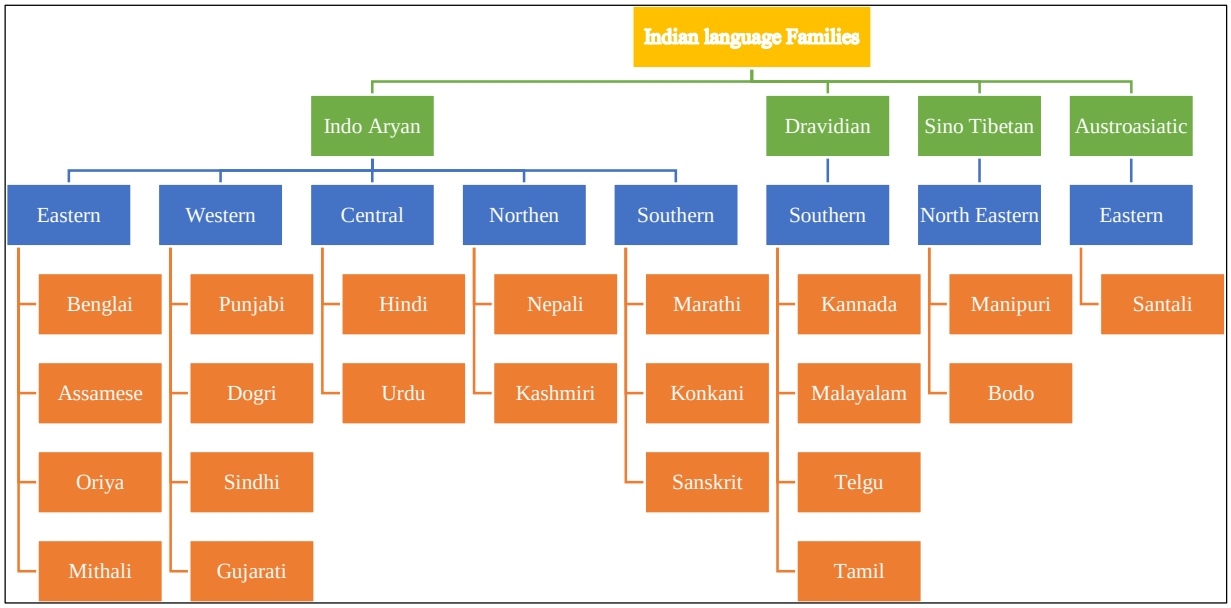


Figure 2.2: Indian language families (Chand, 2016)

Kannada and Malayalam. These languages are mainly used in the southern part of the Indian subcontinent. Besides India the Dravidian language family is also used in northeastern Sri Lanka, Nepal, Pakistan, and Bangladesh (Chand, 2016). Dravidian languages cover around 24% of the total Indian population.

(iii) The Sino-Tibetan language family is older than the Indo-Aryan language family, and it is referred to as Kiratas in Sanskrit literature. The languages of the Sino-Tibetan language family have three major sub-divisions: The Tibeto Himalayan, the north Assamese, and the Assam–Myanomari (Burmese). The Manipuri and Bodomain are the main languages of the Sino-Tibetan language family.

(iv) The Austroasiatic language family is the ancient language family that exists even before the arrival of Aryans. Santali is the most spoken language of this family. In the most archaic Sanskrit literature of India, it is referred to as Nisadas. The Austroasiatic language family has primarily spoken in central, eastern and north-eastern regions of India. However, only 2% of the population of India spokes Austroasiatic and Sino-Tibetan language families.

The major work for the WSD has been done in the English language as well as for European, Chinese, Arabic, *etc*., languages. Minimal work has been reported for Hindi, Bengali, Punjabi and Marathi Indian languages. The main reasons for limited research work in the field of the WSD for Indian languages are lack of knowledge bases, corpus, senes tagged datasets and the linguistic tools for Indian languages. However, the need for interlingual resources and accessibility to computational resources has motivated the researchers to work for the WSD

for Indian languages. Indian states have a policy of usage of three languages that are a state language, Hindi and English language. Thus, machine translation and cross-lingual applications are highly relevant in the Indian context.

2.3.1 Evolution of WSD for Indian languages

The first work in the field of WSD for Indian languages has been reported by (Unnikrishnan & P, 2002) in the year 2002. The authors have performed the WSD for the Hindi language. However, the development and implementation of Indo WordNet by (Bhattacharya, 2010) have given a major impetus to the research field of WSD for Indian languages. The year-wise evolution of the WSD for Indian languages has been depicted in Figure 2.3. The quality research work of WSD for the Indian languages has started in the year 2009 when researchers have reported studies of WSD for Hindi, Kannada and Tamil languages. Later on, the researchers have explored the WSD for other Indian languages such as Marathi, Tamil, Punjabi and Bengali, *etc*. Most of the work in the WSD area for six languages (Assamese, Bengali, Manipuri, Marathi, Nepali and Telugu) has been reported in the year 2013 and 2014. In the Indian context, the WSD for 18 official languages has developed and is still evolving.

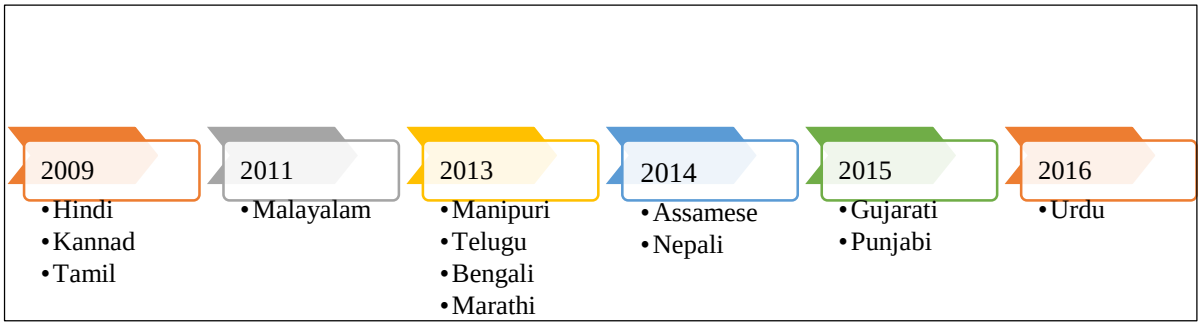


Figure 2.3:Evolution of the WSD for Indian languages

2.4 Extraction Outcomes

This research aims to analyse the research studies in the field of WSD for Indian languages. This systematic analysis has been performed by adopting the research questions as stated in Table 2.1. The outcome of this analysis has been used to answer the research questions outlined in Table 2.1. The following subsections map the answers to the stated research questions.

2.4.1 Publication Timeline of the WSD for Indian Languages

The RQ1 has been answered by depicting the publication timeline of the collected papers. Figure 2.4 shows a graph of yearly percentage publications of the WSD for Indian languages. The exhaustive analysis of research publications shows that most of the research studies on the WSD for Indian languages are published in a wide variety of conference proceedings and

journals. Table 2.3 also suggests that the research studies of the WSD for Indian languages have been published in a variety of e-database resources. It has been observed that the majority of research articles on the WSD for Indian languages are published in conferences and followed by journals.

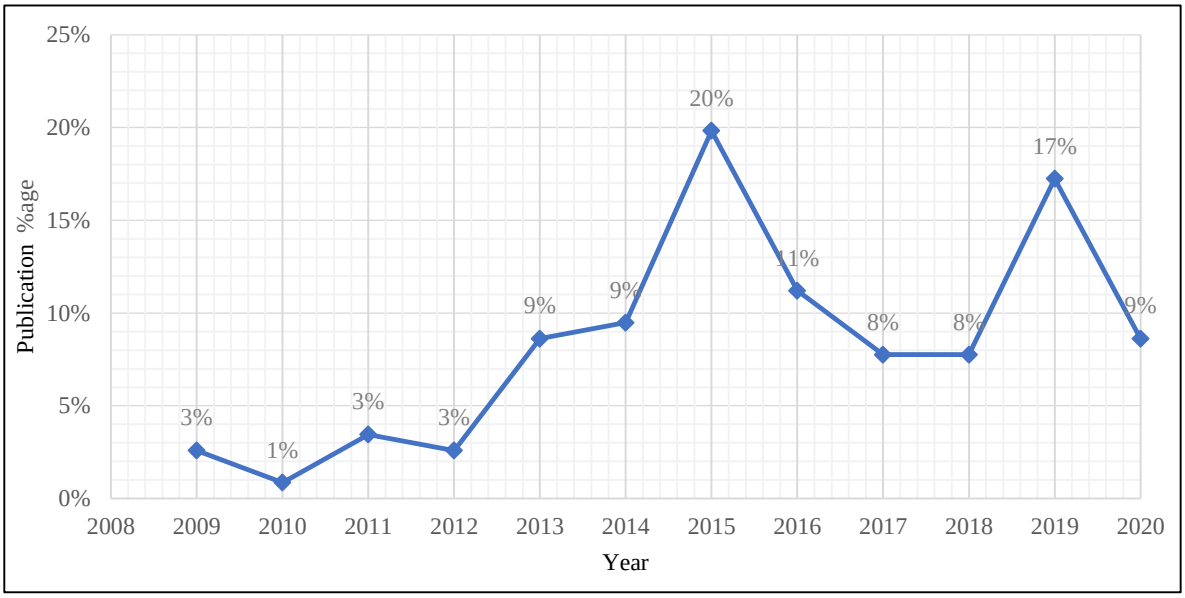


Figure 2.4:Trends in the publications of the WSD for Indian languages

The trend line of the year-wise publication of papers in Figure 2.4 shows that WSD for Indian languages is the emerging area of research.

2.4.2 WSD for Indian Languages

A thorough description of the research work on the WSD for Indian languages has presented alphabetically in this section.

(i) **Assamese**

(Borah, et al., 2014) have experimented with a Naïve Bayes (NB) classifier on the sense tagged dataset of 25 Assamese words to perform the Assamese WSD. (Kalita & Barman, 2015) have applied a knowledge base approach (walker algorithm) for the disambiguation of ambiguous Assamese words by examining the categories that are the senses of the words. The authors have modified Assamese WordNet to extract the categories of words. (Sarmah & Sarma, 2016a) have implemented the Assamese WSD based on the NB classifier using the iterative learning method. The authors have explored 160 ambiguous Assamese words and reported improved performance of the NB classifier for WSD. (Sarmah & Sarma, 2016b) have experimented with a supervised classifier Decision Tree (DT) to perform Assamese WSD. The authors have explored lexical features to disambiguate ten ambiguous Assamese words and reported an average F-measure of 0.611. (Sarmah & Sarma, 2018)

have explored Assamese language prerequisites to implement the Assamese WSD using the NB and DT supervised classifiers. In their thesis work, the authors have reported the comparative analysis between the NB and DT classifiers for the Assamese WSD and have evaluated the impact of the Assamese WSD on the machine translation task. (Borah, et al., 2019) have implemented the NB classifier using the syntactic and semantic features for the Assamese WSD and have reported that semantic features enhance the performance of the NB classifier for the Assamese WSD.

(ii) **Bengali**

(Das & Sarkar, 2013) have explored a graph-based unsupervised approach to derive the sense clusters for the ambiguous Bengali words from a Bengali raw corpus. The authors have applied a vector space approach to map the sense clusters to Hindi translations of the target Bengali word, and these tagged clusters are used to predict the translation of the target Bengali word in the test instance. (Pal, et al., 2015a) have applied the NB classifier on the sense tagged dataset of a single Bengali word to determine its senses. The authors have prepared a dataset of the word from the TDIL Bengali corpus and have selected its senses from the Bengali WordNet. (Pandit & Naskar, 2015) have experimented with a supervised k Nearest Neighbour (k-NN) approach on the sense annotated dataset of 25 Bengali words that has produced an average accuracy of 71%. (Pal, et al., 2015b) have demonstrated the effect of lemmatization on the Bengali WSD. The authors have applied the supervised NB classifier on both the plain and the lemmatized corpus of Bengali ambiguous words and have shown that the NB classifier has yielded better accuracy with lemmatized Bengali corpus. (Haque & Hoque, 2016) have applied a dictionary-based overlap approach to resolve the ambiguity of Bengali ambiguous words. The author's algorithm has assigned the senses to the detected Bengali ambiguous word from the test sentences using a Bengali parser. (Chakrabarty & Garain, 2016) have developed a lemmatization algorithm for the Bengali (BenLem) language and have investigated its effectiveness for the Bengali WSD. The authors have reported the performance enhancement of Bengali WSD using BenLem. (Pal & Saha, 2016) have reported an improvement in the performance of the NB classifier for the Bengali WSD after the inclusion of the test dataset into the training dataset for every next step of the prediction. (Pal & Saha, 2017) have applied a clustering-based unsupervised approach on the Bengali corpus created by TDIL. The authors have manually annotated the sense to the segregated clusters of the target word and have applied cosine similarity measure to determine the sense of the test instance. (Pal, et al., 2017) have applied the context expansion and synset

analysis of the Bengali WordNet to improve the traditional Lesk algorithm for the Bengali WSD. (Nazah, et al., 2017) have applied the NB and Artificial Neural Network (ANN) classifiers to disambiguate Bengali words and have registered an average accuracy of 82%. (Kaysar & Khan, 2018) have applied an unsupervised algorithm (Apriori Algorithm) to find the association rules which have been used to classify Bengali words.

(Sau, et al., 2019) have implemented the WSD for the Bengali language in two steps. In the first step, the authors have induced the senses of the ambiguous words using a raw Bengali corpus. In the second step, the authors have tested the induced senses on ten ambiguous Bengali words. (Kaysar, et al., 2019) have applied a Frequent Pattern (FP-Growth) algorithm to perform the Bengali WSD and concluded that the FP-Growth algorithm has outperformed the Apriori algorithm while performing the Bengali WSD. (Sengupta, et al., 2019) have reported an improvement in the Bengali WSD by the application of the sense induction approach on the mono and parallel Bengali corpus. The baseline k-means algorithm has been used by the authors in their comparative research. (Pal & Saha, 2019) have adopted an extension to the unsupervised clustering approach by including Principal Component Analysis (PCA) and context expansion approach with PCA for the Bengali WSD. The authors have reported the accuracy of 52% and 62% by including PCA and context expansion with PCA approaches for the Bengali WSD. (Pal, et al., 2019) have investigated the ANN, DT, NB, Support Vector Machine (SVM) classifiers for the Bengali WSD by applying for two extensions, namely inclusion of lemmatization process and bootstrapping of the operational process. The authors have stated that both the extensions have improved the accuracy of all the classifiers for the Bengali WSD. (Das Dawn, et al., 2020) have presented a comprehensive survey of Bengali WSD techniques and the existing datasets of the Bengali language.

(iii) **Gujarati**

(Desai, et al., 2015) have applied a simplified Lesk algorithm a knowledge base approach for the Gujarati WSD. (Vaishnav, 2017) has experimented with a genetic algorithm using the Guajarati WordNet as a lexical resource. (Vyas & Ganatra, 2019) have implemented an autoencoder deep learning supervised approach on the term frequency-inverse document frequency features of Gujarati words selected from the Gujarati WordNet. (Vaishnav & Sajja , 2019) have applied the genetic algorithm to perform the Gujarati WSD.

(iv) **Hindi**

(Sinha, et al., 2004) have performed the automatic WSD for the Hindi language by exploring the Hindi WordNet. The authors have applied a knowledge based Lesk algorithm

to perform the WSD on the Hindi ambiguous noun words. (Khapra, et al., 2008) have performed the domain-specific iterative sense disambiguation of English, Hindi and Marathi (nouns, adjectives and adverbs) words by applying the knowledge based graph structures and corpus biases algorithms. The authors have reported F1 score of 65% for the health and tourism domains of the English, Hindi and Marathi languages. (Mishra, et al., 2009) have evaluated a DL algorithm on the 20 ambiguous Hindi noun words after removing the stop words and applying stemming on the dataset. (Khapra, et al., 2009) have performed all word disambiguation for the Hindi, Marathi, Bengali and Tamil languages using iterative WSD and modified PageRank approaches. The authors have projected the corpus and WordNet parameters of source language on target language and have applied a novel synset based dictionary for the sense classification. (Rastogi & Dwivedi, 2011) have investigated the performance of the Hindi WSD based on the highest sense count approach. The authors have experimented with 100 queries on Google, Raftaar and Guruji search engines and have reported an improvement in the information retrieval of the Hindi text using the Hindi WSD.

(Mishra & Siddiqui, 2012) have investigated Yarowsky's algorithm for the Hindi WSD by applying the Hindi WordNet to label the clusters of Hindi text. The authors have reported maximum precision of 61.7% in their research work. (Singh & Siddiqui, 2012) have evaluated a knowledge based Lesk algorithm on 10 Hindi words by considering the parameters such as the context window size, stop word removal and the stemming of words. (Bala, 2013a) has developed a WSD tool using a knowledge based overlap approach that has been evaluated on 100 ambiguous Hindi words. (Bala, 2013b) has performed Hindi WSD using a selectional restriction, a knowledge based approach on the Hindi WordNet. (Yadav & Vishwakarma, 2013) have mined the association rules of the different senses of the Hindi ambiguous words from the Hindi WordNet and have explored these rules to determine the sense of the Hindi ambiguous words. (Jain, et al., 2013) have performed the all word Hindi WSD by evaluating the graph of words in a sentence using graph-based clustering measures, namely denseness, graph randomness and edge density. (Singh, et al., 2013) have applied the semantic similarity measure for the Hindi WSD by extracting the semantic similarity of words from the Hindi WordNet. The authors have evaluated their similarity method on the 20 polysemous Hindi noun words and have reported an accuracy of 60.65%.

(Singh, et al., 2014) have leveraged a dictionary-based Hindi WSD module in Hindi language translation. The authors have applied n-gram smoothing models in the Hindi

WSD module. (Sawhney & Kaur, 2014) have applied the modified Lesk algorithm that considered a dynamic context window around the target ambiguous Hindi word to accomplish the Hindi WSD. (Singh, et al., 2014) have investigated eleven features of the 60 Hindi ambiguous noun words to disambiguate them using a supervised NB classifier. The authors have concluded that the addition of rich features has significantly improved the performance of the NB classifier. (Jain & Lobiyal, 2014) have performed the Hindi WSD with the help of a graph extracted from the lexical-semantic relations defined in the Hindi WordNet. The authors have determined the senses of all words in a Hindi sentence by estimating the minimum cost of spanning trees corresponding to the semantic graph of different senses of the words.

(Singh & Siddiqui, 2015a) have investigated the role of hypernym, hyponym, holonym and meronym relations of the Hindi WordNet in the disambiguation of 60 Hindi polysemous words. The authors have reported an improvement of 12.09% in the average precision over the baseline precision of the 60 ambiguous words when all relations have been considered. (Jain & Lobiyal, 2015a) have performed the Hindi WSD by employing network agglomeration, an unsupervised approach on Hindi sentences. (Nandanwar, 2015) has explored the sense graphs derived from the Hindi WordNet to disambiguate the Hindi ambiguous words. (Bhingardive, et al., 2015) have prepared the word embedding from the Hindi language corpora and have compared it with the sense embedding of the Hindi word to obtain its predominant sense by similarity measure method. The authors have reported an improvement in the Hindi WSD over the WordNet first sense baseline. (Singh & Siddiqui, 2015b) have explored the role of semantico-syntactic relationships between verbs and nouns or pronouns called "*Karaka*" for the Hindi WSD by utilizing "*vibhaktis*". The authors have applied the NB and conditional probability classifiers for the Hindi WSD.

(Singh & Siddiqui, 2015c) have compared the three WSD algorithms, corpus-based Lesk, co-occurrence conditional probability and the information model to perform the corpus-based Hindi WSD. The authors have utilized four metrics Term Frequency (TF), Inverse Document Frequency (IDF), Term Frequency-Inverse Document frequency (TF-IDF) and Content Words (CW) to compute the weights of matching words. The authors have reported overall maximum precision of 85.87% for the corpus-based Lesk algorithm using the TF-IDF weighting metric. (Garg & Mittal, 2015) have developed a two-stage process for the Hindi WSD. In the first stage, the authors have applied the modified Lesk algorithm on nine ambiguous Hindi words and in the second stage, the output of the first stage has been fed to the SVM classifier to disambiguate these ambiguous Hindi words. (Patel, et al., 2015)

have applied hierarchical clustering algorithm using cosine, Jaccard and Dice similarity measures to perform the Hindi WSD. The authors have overlapped the clusters prepared from the Hindi corpus with the Hindi WordNet and reported an improvement in Hindi WSD. (Jain & Lobiyal, 2015b) have prepared an extended version of the Hindi WordNet called the fuzzy Hindi WordNet. The authors have explored the fuzzy graph connection measures from fuzzy Hindi WordNet to disambiguate the Hindi words. (Tayal, et al., 2015) have developed an unsupervised approach for the Hindi WSD. The authors have employed a fuzzy C-means clustering algorithm on the Hindi text vectors to perform the Hindi WSD. (Sarika & Sharma, 2016) have applied the cosine similarity approach to perform the Hindi WSD on 90 ambiguous words and have reported an average precision and recall of 78.99 % and 72.58 %. (Kumar, et al., 2016) have evaluated three supervised approaches, NB, Support Vector Machine (SVM) and Neural Network (NN), on the five feature sets (bag of words, collocations, bigram, trigram, PoS categories) of the Hindi ambiguous words to perform the Hindi WSD. (Gautam & Sharma, 2016) have applied the Lesk algorithm on the bigrams and trigrams words extracted from the corpus containing 15 polysemic Hindi verb words. The authors have reported 52.98% precision for bigram words and 37.04% precision for trigram words.

(Bhingardive & Bhattacharyya, 2017) have explored the IndoWordNet features such as linked WordNets and lexico-semantic relations to perform the WSD. The authors have applied the expectation maximization algorithm in the bilingual approach for the Hindi WSD and have used IndoWordNet to find the most frequent sense using embedding features. (Kumar & Yadav, 2018) have implemented the sense based machine translation from the Hindi to the Tamil language. The authors have determined the senses of the Hindi ambiguous words using the Latent Semantic Analysis (LSA) approach and then have applied statistical machine translation. (Athaiya, et al., 2018) have experimented with a genetic algorithm that has explored the neighbouring words of the target Hindi word for the Hindi WSD.

(Sharma & Joshi, 2019)have developed a Hindi WSD tool based on the knowledge based overlap approach. The authors have explored the WordNet of the Hindi language in their WSD tool and have reported an accuracy of about 71.4%. (Jain & Lobiyal, 2019a) have devised Fuzzy Hindi WordNet (FHWN), an extension of the Hindi WordNet to perform the Hindi WSD. The authors have explored the fuzzy semantic relations using the central tendency measures and have deduced the winner sense with the highest measure score. (Jain & Lobiyal, 2019b) have applied an epidemic spreading based model to perform the

Hindi WSD and have reported that expected forces yield better results than the other centrality measures. (Kumari & Lobiyal, 2020) have extracted continuous bag of words and Skip-gram Word2Vec embedding features of the Hindi ambiguous words and have leveraged the cosine similarity measure to find the proximity of the correct sense with the ambiguous word. (Tripathi, et al., 2020) have applied Hindi WordNet and the score based modified lesk algorithm to disambiguate the Hindi language. (Chakrawarti, et al., 2020) have incorporated Hindi WSD to perform phrase based statistical machine translation of the Hindi poetries into English. The authors have suggested that Hindi WSD based on the lesk algorithm in conjunction with Hindi WordNet has imporved the experimental results of statistical machine translation of the Hindi poetries into English.

(v) **Kannada**

(Parameswarappa & Narayana, 2011a) have developed an integrated Kannada WSD system that has also performed the corpus building and dictionary building tasks. The rule-based WSD system developed by the authors disambiguates all the ambiguous words belonging to the different lexical categories such as noun, verb, adverb and adjective. (Parameswarappa & Narayana, 2011b) have explored the compound word clues and syntactic features to developed a targeted WSD for the Kannada language based on the NB supervised approach. The authors have applied a Kannada shallow parser to extract syntactic features. (Parameswarappa & Narayana, 2011c) have explored the association rules to disambiguate all the content words like nouns, verbs, adjectives, adverbs in an unrestricted Kannada text. (Parameswarappa & Narayana, 2012) have performed Kannada verb WSD by exploring the argument structure of verbs. The authors have extracted the morphological and syntactic features using Kannada shallow parser and experimented with their Kannada WSD on 30 most polysemous Kannada verbs. (Parameswarappa, et al., 2013) have explored the DL classifier to disambiguate all content words such as nouns, verbs, adverbs, adjectives in the Kannada text using collocation features. (Rao & Kallimani, 2016) have prepared the Kannada polysemy word analyser, which determines the ambiguous word and its senses with the help of a Kannada shallow parser and a Kannada dictionary. (Shashank & Kallimani, 2017) have combined the glosses of neighbouring words of the target Kannada word with their respective PoS tags to perform the Kannada WSD. (Kumar, et al., 2018) have exploited three adaptations of the Lesk algorithm to disambiguate 10 Kannada ambiguous words using Kannada WordNet.

(vi) **Malayalam**

(Haroon, 2010) has implemented the Malayalam WSD using an overlap algorithm, a knowledge based approach. (Jisha, et al., 2015) have experimented with maximum entropy, a supervised approach to disambiguate a Malayalam ambiguous word. (Junaid, et al., 2015) have explored a hybrid approach that combines SVM and corpus-based overlap approach to perform the Malayalam WSD. The authors have reported the appropriateness of multiclass SVM for Malayalam WSD. (Gopal & Haroon, 2016) have reported 95% reliability while using the NB classifier for the Malayalam WSD on the corpora of 100K words. (KP, et al., 2016) have prepared the seed sense clusters of the ambiguous Malayalam words by selecting sense details from the Datuk corpus, a sense tagged of the Malayalam language. The authors have enriched the seed clusters by including collocation, Word2Vec, and co-occurring features of words then have predicted the sense of the target word using the similarity between the test instance and the sense clusters. (Junaida, et al., 2017) have applied Conditional Random Field (CRF) and Margin Infused Relaxed Algorithm (MIRA) on the unigram, bigram and their combination with PoS tag features selected from the sense tagged dataset of Malayalam words to perform Malayalam WSD. (Mohan Raj, et al., 2019) have processed Malayalam corpus using clustering and deep learning approaches to perform WSD. The authors have created a collocation dictionary from the clusters of different senses of the ambiguous words to perform the WSD. The authors also have applied the data-driven recurrent neural network, a deep learning approach to disambiguate homonymous Malayalam noun words.

(vii) **Manipuri**

(Singh, et al., 2014) have manifested Manipuri WSD using the supervised Classification and Regression Trees (CART) classifier based on context and positional features. The authors have reported an accuracy of 71.75% when their Manipuri WSD has been trained on 1600 ambiguous words and tested on 400 ambiguous words.

(viii) **Marathi**

(Khapra, et al., 2008) have explored the trilingual Hindi, Marathi and English corpus biases along with their respective WordNet graph structures to perform the domain-specific WSD (Health and Tourism). (Khapra, et al., 2009) have projected the WordNet and corpus parameters of resource sufficient language (Hindi) to resource scarce language (Marathi, Bengali and Tamil), they have applied the iterative WSD and PageRank algorithms for the Marathi WSD. The authors have reported a 72% F-score by the iterative WSD for Marathi language. (Dhopavkar, et al., 2014) have devised and applied the entropy-based rules for

the Marathi WSD. (Dhopavkar, et al., 2015) have formulated the word and sentence rules for the disambiguation of Marathi text. (Zungre & Dhopavkar, 2016) have prepared a decision graph from the Marathi WordNet and explored it for the Marathi WSD. (Patil, et al., 2020) have provide the information of Marathi language resources that can be used in Marathi WSD.

(ix) **Nepali**

(Shrestha, et al., 2008) have studied the impact of the morphological analyser on the performance of the Lesk algorithm when applied for the Nepali WSD. (Dhungana & Shakya, 2014) have implemented a modified, adapted Lesk algorithm for the disambiguation of 348 words, including 59 polysemy Nepali words, and have reported an average accuracy of 88.05%. (Roy, et al., 2014) have evaluated the conceptual distance and semantic graph approaches on the Nepali WordNet for disambiguation of the ambiguous Nepali words.

(x) **Punjabi**

The n-gram language modelling technique has been applied by (Josan & Lehal, 2008) for the disambiguation of the Punjabi language words. The authors have concluded that the smaller *n* for the n-gram model has yielded a better disambiguation rate. (Rana & Kumar, 2015) have applied a knowledge based Lesk algorithm on 30 ambiguous Punjabi noun words extracted from the Punjabi WordNet. (Kaur & Singh, 2015) have applied a customized Lesk algorithm a knowledge based approach on the ambiguous Punjabi words selected from the medical domain. (Singh & Singh, 2015) have applied an efficient modified Lesk algorithm on the corpus of 15 Punjabi ambiguous noun words. (Walia, et al., 2017) have reported higher accuracy produced by a supervised NB classifier when evaluated on 100 sense tagged words selected from Punjabi corpus. (Walia, et al., 2018) have experimented with a supervised k-NN algorithm on the sense tagged dataset of 100 ambiguous Punjabi words. (Singh & Kumar, 2018) have developed an independent Punjabi WSD using the NB classifier. The authors have experimented with the bag of words and collocation features extracted from the sense tagged dataset of 150 Punjabi words and have reported that the bag of words features yield better results than collocation features. (Walia, et al., 2018) have explored the three supervised classifiers NB, k-NN and DT that have learned contextual rules from the sense-annotated examples of the ambiguous Punjabi words to perform the Punjabi WSD. (Walia, et al., 2019) have performed a case-based reasoning approach to determine the sense of ambiguous Punjabi words. We (Singh & Kumar, 2019) have manifested the application of NB, DL, k-NN, SVM, Random Forest

(RF), and DT six supervised and two deep learning Multi Layer Perceptron (MLP), and Long Short Term Memory (LSTM) classifiers on lexical, syntactic and semantic features extracted from the sense tagged dataset of 150 ambiguous Punjabi words. Deep learning approaches MLP and LSTM have applied by us (Singh & Kumar, 2019) on the sense tagged dataset of 66 Punjabi words and have reported that the deep learning approaches outperform the machine learning approaches for the Punjabi WSD. This thesis covers some content of our publications in the later chapters with a lot more details.

(xi) **Tamil**

(Sankaran & Vijay-Shanker, 2003) have explored the influence of morphology on the Tamil WSD and have manifested the DL classifier for the Tamil WSD. (Baskaran & Vaidehi, 2004) have segregated the sense cluster of the ambiguous words from the training corpus. The authors have manually applied the extracted prominent collocations to determine the senses of the Tamil ambiguous words. (Rajendran, et al., 2013) have discussed the application of the collocations words in conceptual graph dictionary and the cluster approach for the Tamil WSD. (Anand Kumar, et al., 2014) have demonstrated the supervised framework using an SVM classifier for the Tamil WSD based on context features. (Padmamala, 2015) has developed a rule-based Tamil WSD and a syntactic parser using semantic information to perform translation of Tamil words to English words using the word level knowledge based translation approach. (Iswarya & Radha, 2017) have reported a two-step Tamil WSD. In the first step, the PoS tagger has been applied to find the sense of the Tamil ambiguous word based on the PoS category and in the second step, if it fails, a clustering approach has been followed to select the sense of the Tamil ambiguous word.

(xii) **Telugu**

(Palanati & Kolikipogu, 2013) have developed a Telugu WSD using the DL algorithm and evaluated it with the Telugu information retrieval task. The authors have reported that the Telugu WSD has improved the performance of the information retrieval task. (Sreedhar, et al., 2014) have framed and applied rules to disambiguate compound two-word and three-words. (Durga Prasad, et al., 2018) have applied statistical approaches to disambiguate the Telugu nouns and verbs words. The machine learning and a graph-based approach for the Telugu WSD has been experimented by (Koppula, et al., 2019). (Durga Prasad, et al., 2019) have explored the co-occurrence features selected by the n-gram method for the Telugu WSD. The authors have reported the F-measure of 0.94 for the Telugu WSD when tested

on the Telugu corpus by CIIL. (Koppula, et al., 2020) have explored the knowledge based approach to disambiguate the Telugu text.

(xiii) **Urdu**

(Naseer & Hussain, 2009) have performed the first Urdu WSD using a supervised Bayesian Network classification approach. The authors have concluded that the large context window size has yielded the best results. The NB and SVM classifiers have experimented on the sense tagged corpus of Urdu words for the Urdu WSD by (Arif, et al., 2016). Three machine learning classifiers, BN, SVM and DT have been evaluated and compared by (Abid, et al., 2018) for the Urdu WSD and the authors have reported the best F-measure 0.711 for the Bayesian network classifier. (Saeed, et al., 2019) have prepared a corpus for Urdu lanague and have experimented it for all-word Urdu WSD. (Saeed, et al., 2019) have performed the Urdu WSD with the help of the Urdu Lughat dictionary a manually crafted dictionary for 50 Urdu ambiguous words. The authors have experimented with five supervised classifiers, NB, SVM, kNN, DT, MLP, in their research. The authors have prepared a standard all word Urdu corpus having 856 instances with 5042 sense tagged words using the Urdu Lughat dictionary. The authors have experimented with n-gram WSD models based on six semantic similarity approaches (Jaccard, Overlap, Dice, Euclidean distance and Cosine similarity) on the all word sense tagged Urdu corpus.

Table 2.4 summarizes the different approaches, type of WSD (Target or All-words), resources, datasets and evaluation parameters (Accuracy (A), Precision (P), Recall (R), F-Measure (F/F1)) used to perform the WSD for the Indian languages.

Table 2.4: Summarized details of the WSD for Indian languages

Language Author(s)	**Approach Classifier(s)**	**WSD Type**	**Resource(s)**	**Dataset size**	**Evaluation measures**
Assamese (Borah, et al., 2014)	Supervised NB	Target WSD	Assamese WordNet	25 words	F1:86%
Assamese (Kalita & Barman, 2015)	Knowledge based Walker algorithm	All-words WSD	Assamese WordNet	82 sentences	Set1- P:81.25%, R:68.42% Set2- P: 88.88%, R:36%
Assamese (Sarmah & Sarma, 2016a)	Supervised NB	Target WSD	Assamese WordNet	10 words	F:61.10%
Assamese (Sarmah & Sarma, 2016b)	Supervised DT	Target WSD	Assamese WordNet	Tagged 3.5K, 160 words	A:78%
Assamese (Sarmah & Sarma, 2018)	Supervised NB,DT	Target WSD	Assamese WordNet	Tagged 4.5K, 60 words	A:78%

Language Author(s)	**Approach Classifier(s)**	**WSD Type**	**Resource(s)**	**Dataset size**	**Evaluation measures**
Assamese (Borah, et al., 2019)	Supervised NB	Target WSD	Assamese WordNet	135 words	F1:91.11%
Bengali 2013 (Das & Sarkar, 2013)	Unsupervised Graph based clustering	All-words WSD	Hindi WordNet, Bengali sense dictionary, Bengali-Hindi News corpora	10 words	A: 72.79%
Bengali 2015 (Pal, et al., 2015a)	Supervised NB	Target WSD	Bengali WordNet, TDIL Bengali corpus 271102 sentences	one word	A:84%
Bengali 2015 (Pandit & Naskar, 2015)	Supervised kNN	Target WSD	Train dataset 250 sentences, 100 test sentences	25 words	A:71%
Bengali 2015 (Pal, et al., 2015b)	Supervised NB	Target WSD	TDIL Bengali corpus 271102 sentences	4 words	A: 85%
Bengali 2016 (Pal & Saha, 2016)	Supervised NB	Not specified	TDIL Bengali corpus 271102 sentences	Not specified	A:83%
Bengali (Chakrabarty & Garain, 2016)	Knowledge based Lesk algorithm	Not specified	Bengali WordNet	10 words	R:54.90%
Bengali (Haque & Hoque, 2016)	Knowledge based Overlap	All-words WSD	Bengali Dictionary	500 sentences	A:82.40%
Bengali (Pal & Saha, 2017a)	Unsupervised Clustering	Target WSD	Bengali WordNet	one word	A:63%
Bengali (Pal, et al., 2017b)	Knowledge based Lesk algorithm	Target WSD	Bengali WordNet, Bengali PoS tagged corpus	485 sentences of 9 ambiguous words	A:75%
Bengali (Nazah, et al., 2017)	Supervised NB, ANN	Target WSD	Not specified	160 sentences of 50 ambiguous words	A:82%
Bengali (Kaysar & Khan, 2018)	Unsupervised Apriori	All-words WSD	Not specified	1600 sentence patterns	Not specified
Bengali (Sau, et al., 2019)	Unsupervised Sense induction	All-words WSD	TDIL corpora	10 words	A:63.71%
Bengali (Kaysar, et al., 2019)	Unsupervised	All-words WSD	Bangla Database	1000 sentences	A:80%

Language Author(s)	**Approach Classifier(s)**	**WSD Type**	**Resource(s)**	**Dataset size**	**Evaluation measures**
	FP growth for association rules				
Bengali (Sengupta, et al., 2019)	Unsupervised Sense induction	Target WSD	Bangla Database of 111 sentences	15 words	P:81%, R:77%, F:72%
Bengali (Pal & Saha, 2019)	Unsupervised Maximum Entropy	Target WSD	TDIL corpora Bengali WordNet	10 words	A:61%
Bengali (Pal, et al., 2019)	Supervised DT,SVM,ANN,NB	Target WSD	TDIL corpora Bengali WordNet	13 words	A(DT):70.92% A(SVM):79.15% A(ANN):79.53% A(NB):83%
Gujarati (Desai, et al., 2015)	Knowledge based Lesk algorithm	Not Specified	Gujarati WordNet	Not Specified	A:93%
Gujarati (Vaishnav, 2017)	Knowledge based Genetic algorithm	Not Specified	Gujarati WordNet	Not Specified	Not Specified
Gujarati (Vyas & Ganatra, 2019)	Supervised Deep learning	Not Specified	Gujarati WordNet, Wikipedia (27,800 pages	Not Specified	Not Specified
Gujarati (Vaishnav & Sajja , 2019)	Knowledge based Genetic algorithm	Not Specified	Gujarati WordNet	Not Specified	Not Specified
Hindi (Sinha, et al., 2004)	Knowledge base Lesk algorithm	Not Specified	Hindi WordNet	Noun words	A:70%
Hindi (Khapra, et al., 2008)	Knowledge base Graph algorithm	All-words WSD	Hindi, Marathi WordNet	Health: 4280 Tourisim:1962	F:65%
Hindi (Mishra, et al., 2009)	Unsupervised	All-words WSD	Hindi WordNet, Training set: 1856, Test set: 1641 sentences	20 words	Accuracy
Hindi (Khapra, et al., 2009)	Knowledge base PageRank and iterative WSD	All-words WSD	Hindi, Marathi, Bengali and Tamil WordNet,	Hindi (health, Tourism): 50890, 29631, Marathi (health, Tourism): 32694, 8540 Bengali (Tourism): 9435, Tamil(Tourism): 17868	F1:75%

Language Author(s)	**Approach Classifier(s)**	**WSD Type**	**Resource(s)**	**Dataset size**	**Evaluation measures**
Hindi (Rastogi & Dwivedi, 2011)	Unsupervised	Target WSD	Not Specified	5 words	Not Specified
Hindi (Mishra & Siddiqui, 2012)	Unsupervised Decision List	Target WSD	Hindi WordNet	40 words	P:61.70%
Hindi (Singh & Siddiqui, 2012)	Knowledge base Lesk algorithm	Target WSD	Hindi WordNet	10 words	A:54.81%
Hindi (Bala, 2013a)	Knowledge base Overlap with thesauri	Target WSD	Hindi WordNet	100 words	P:58.82% R:50%
Hindi (Bala, 2013b)	Knowledge base Selectional restriction	Target WSD	Hindi WordNet, 100 verb words, test dataset 5236 words	5 words	A:66.92%
Hindi (Yadav & Vishwakarma, 2013)	Knowledge base Association rules	Target WSD	Hindi WordNet	3 words	A:72%
Hindi (Jain, et al., 2013)	Knowledge base Graph connectivity and clustering	All-words WSD	Hindi WordNet	1200 words	Set1 A: 60.25% Set2 A: 41.25% Set3 C:54.81%
Hindi (Singh, et al., 2013)	Supervised Semantic Similarity	Target WSD	Hindi WordNet	20 words	A:60.65%
Hindi (Singh, et al., 2014)	Knowledge base Dictionary based overlap with N-grams	Target WSD	Not Specified	10 words	Set1 A: 65% Set2 A:67%
Hindi (Sawhney & Kaur, 2014)	Knowledge base Modified Lesk algorithm	Target WSD	Not Specified	10 words	Precision
Hindi (Singh, et al., 2014)	Supervised NB	Target WSD	Hindi WordNet, Training dataset: 7506 sentences	60 words	Set1 P: 77.52%, Set2 P: 86.11% Set3 P: 56.49%
Hindi (Jain & Lobiyal, 2014)	Unsupervised Graph based minimum spanning tree	Not Specified	Hindi WordNet	Not Specified	Not Specified
Hindi (Singh & Siddiqui, 2015a)	Knowledge base Overlap	Target WSD	Hindi WordNet	Training dataset: 7506 sentences of 60 words	P:56.76% R:51.32%
Hindi (Jain & Lobiyal, 2015a)	Unsupervised Graph based Network agglomeration	All-words WSD	Hindi WordNet,	Hindi (Health, Tourism): 50890, 29631, dataset	Set 1 (H) F:50.20% Set 2 (H) F:43.05% Set 1 (T) F:53.01% Set 2 (T) F:44.02%
Hindi (Nandanwar, 2015)	Knowledge base Graph based algorithm	Target WSD	Hindi WordNet,	one word	Not Specified

Language Author(s)	Approach Classifier(s)	WSD Type	Resource(s)	Dataset size	Evaluation measures
Hindi (Bhingardive, et al., 2015)	Unsupervised Word embedding	All-words WSD	Hindi WordNet	Tr: 80,000 polysemous noun words	Set1: P:62.43%, R: 61.58%, F1:62.00% Set2: P:61.73% R:59.31% F1:60.49%
Hindi (Singh & Siddiqui, 2015b)	Supervised Naïve bayes, Conditional probability	Not Specified	Hindi WordNet,	Not Specified	Set1 P: 50.73%, Set2 P: 56.56%
Hindi (Singh & Siddiqui, 2015c)	Knowledge base Lesk, Conditional probability, Information model	Target WSD	Hindi WordNet Training dataset: 7506 sentences	60 words	P(TF-IDF):85.87% P(CP):68.73% P(IM):76.34%
Hindi (Garg & Mittal, 2015)	Supervised Modified Lesk+ SVM	Not Specified	Not Specified	9 words	Not Specified
Hindi (Patel, et al., 2015)	Unsupervised Hierarchical clustering	Not Specified	Hindi WordNet History 246 words Social study 1279 Short story 2672	Not Specified	P (Cosine) :81.64% P (Jaccard) :80.38% P (Dice) : 79.74%
Hindi (Jain & Lobiyal, 2015b)	Knowledge base Fuzzy graphs	All-words WSD	Hindi WordNet Fuzzy WordNet	Hindi sense tagged health corpus 50890 words	F1 Measure
Hindi (Tayal, et al., 2015)	Unsupervised Hyperanaloge and C-mean clustering	All-words WSD	Hindi Wikipedia, and newspapers	Training dataset 3753 words	A: 79.16%
Hindi (Sarika & Sharma, 2016)	Supervised Cosine similarity	Target WSD	Not Specified	90 words	P:78.99% R:72.58%
Hindi (Kumar, et al., 2016)	Supervised SVM, NN,NB	All-words WSD	Hindi Danik Bhaskar and Navbharat Times newspaper	Sense tagged 20,000 sentences, Test data 1000 sentences	A(NB): 28.46% A(NN): 34.18% A(SVM): 37.33%
Hindi (Gautam & Sharma, 2016)	Knowledge base Lesk algorithm	Target WSD	Hindi WordNet	15 words	Set1 P:52.98% R:33.17% Set2 P:37.04% R:18.56%
Hindi (Bhingardive & Bhattacharyya, 2017)	Unsupervised Word embedding	All-words WSD	Hindi WordNet	80,000 noun words	Set1: P: 62.43% R: 61.58%, F1:62.00%

Language Author(s)	**Approach Classifier(s)**	**WSD Type**	**Resource(s)**	**Dataset size**	**Evaluation measures**
					Set2: P:61.73% R:59.31% F1:60.49%
Hindi (Kumar & Yadav, 2018)	Unsupervised LSA	Not Specified	Not Specified	Not Specified	Not Specified
Hindi (Athaiya, et al., 2018)	Knowledge base Genetic Algorithm	Target WSD	Hindi WordNet, (TDIL) Hindi dataset	8 words	Not Specified
Hindi (Sharma & Joshi, 2019)	Knowledge base Overlap algorithm	All-words WSD	Hindi WordNet	Test data 3000 ambiguous sentences	A:71.4%
Hindi (Jain & Lobiyal, 2019a)	Knowledge base Centrality measures	All-words WSD	Fuzzy Hindi WordNet	Hindi sense tagged health corpus 50890 words	A:65.12%
Hindi (Jain & Lobiyal, 2019b)	Unsupervised graph based expected force centrality measure	All-words WSD	Hindi WordNet	Hindi sense tagged health corpus 50890 words	Set1 A:62.86% Set2 A:52.30%
Hindi (Kumari & Lobiyal, 2020)	Unsupervised Word Embedding, similarity measures	Not Specified	Not Specified	Not Specified	A:52%
Kannada (Parameswarappa & Narayana, 2011a)	Supervised NB	Target WSD	Kannada Shallow parser	one word (2153 sentences)	A:71%
Kannada (Parameswarappa & Narayana, 2011b)	Supervised NB	All-words WSD	Kannada Shallow parser, 50000 words Dictionary	500 sentences	Not Specified
Kannada (Parameswarappa & Narayana, 2011c)	Unsupervised Association rules	Not Specified	Not Specified	Not Specified	Not Specified
Kannada (Parameswarappa & Narayana, 2012)	Unsupervised Cluster comparison	Target WSD	Kannada Shallow parser, Kannada electronic dictionary	30 words	Not Specified
Kannada (Parameswarappa, et al., 2013)	Supervised DL	Not Specified	Kannada Shallow parser, 50000 words Dictionary	Not Specified	Not Specified

Language Author(s)	**Approach Classifier(s)**	**WSD Type**	**Resource(s)**	**Dataset size**	**Evaluation measures**
Kannada (Rao & Kallimani, 2016)	Knowledge Base Database search	Not Specified	Shallow parser	28 sentences	A:71.4%
Kannada (Shashank & Kallimani, 2017)	Knowledge Base Lesk algorithm	Not Specified	Kannada WordNet	Not Specified	Not Specified
Kannada (Kumar, et al., 2018)	Knowledge Base Lesk algorithm	Target WSD	Kannada WordNet	10 words	Not Specified
Malayalam (Haroon, 2010)	Knowledge Base Lesk algorithm	Not Specified	Malayalam WordNet	Not Specified	Not Specified
Malayalam (Jisha, et al., 2015)	Supervised Maximum Entropy	Target WSD	Not Specified	one word	P:84.1%, R:61.32%
Malayalam (Junaid, et al., 2015)	Supervised SVM	Target WSD	Not specified	5 words	P:63.05%, R:57.78%, F:57.9%
Malayalam (Gopal & Haroon, 2016)	Supervised NB	Target WSD	100,000 words	one word	A:95%
Malayalam (KP, et al., 2016)	Unsupervised Clustering	Not Specified	Silpa stemmer, Datuk corpus 83,000 Malayalam words	100 sentences	A:72%
Malayalam (Junaida, et al., 2017)	Supervised Conditional Random Field (CRF) and Margin Infused Relaxed algorithm (MIRA)	Target WSD	Not Specified	5 words	MIRA: P:62.59%, R:63.04%, F:59.82%
Malayalam (Mohan Raj, et al., 2019)	Supervised Clustering by k-NN and RNN	Target WSD	Not Specified	one word	A:91%
Manipuri (Singh, et al., 2014)	Supervised DT	Not Specified	Manipuri database of 672 sentences	2000 words	A:71.75%
Marathi (Dhopavkar, et al., 2014)	Supervised Maximum Entropy	Not Specified	Marathi WordNet	Not Specified	A:80%
Marathi (Dhopavkar, et al., 2015)	Unsupervised Genetic algorithm	Not Specified	Not Specified	Not Specified	Not Specified

Language Author(s)	**Approach Classifier(s)**	**WSD Type**	**Resource(s)**	**Dataset size**	**Evaluation measures**
Marathi (Zungre & Dhopavkar, 2016)	Knowledge Base Decision graph	Not Specified	Marathi WordNet	Not Specified	Not Specified
Nepali (Shrestha, et al., 2008)	Knowledge Base Lesk algorithm	Not Specified	Nepali WordNet, Nepali Morph analyser	Not Specified	A:80%
Nepali (Dhungana & Shakya, 2014)	Knowledge Base Adapted Lesk algorithm	Target WSD	Nepali WordNet	348 words	A:88.05%
Nepali (Roy, et al., 2014)	Knowledge Base Overlap (O), Sematic distance and Semantic graph (SS)	All-words WSD	Nepali WordNet	1663 words [Noun (N) 912, Adjective (A)751]	Set(O) A(N): 54%, A(A): 42% Set (SS) A(N): 62%, A(A): 58%
Punjabi (Josan & Lehal, 2008)	Supervised Entropy	Target WSD	Punjabi database of 50000 words	1000 words	Set1:ER:7.96% Set2: ER:3.7%
Punjabi (Rana & Kumar, 2015)	Knowledge Base Lesk algorithm	Target WSD	Punjabi WordNet	30 words	Not Specified
Punjabi (Kaur & Singh, 2015)	Knowledge Base Lesk algorithm	Target WSD	Punjabi WordNet	10 words	Not Specified
Punjabi (Singh & Singh, 2015)	Knowledge Base Enhanced Lesk algorithm	Target WSD	1549 instances of Punjabi WordNet	15 words	Not Specified
Punjabi (Walia, et al., 2017)	Supervised NB	Target WSD	Punjabi corpus by ELDA	3 words	Set1 A:91.01% Set2 A: 98.80% Set1 A: 98.47% Set2 A:100% Set1 A:96.12% Set2 A: 97.57%
Punjabi (Walia, et al., 2018)	Supervised kNN	Target WSD	Punjabi WordNet	8 words	A:71.20%
Punjabi (Singh & Kumar, 2018)	Supervised NB	Target WSD	Punjabi WordNet	150 words	Set1 A:89.62% Set2 A:81.97%
Punjabi (Walia, et al., 2018)	Supervised NB, k-NN, DT	Target WSD	Punjabi WordNet	20 words	Not Specified
Punjabi (Walia, et al., 2019)	Case based interpretation model	Target WSD	Punjabi corpus by ELDA	3words	Set1 A:58.65% Set2 A:51.03% Set3 A:67.10%
Punjabi (Singh & Kumar, 2019)	Supervised	Target WSD	Punjabi WordNet	150 words	A:84%

Language Author(s)	**Approach Classifier(s)**	**WSD Type**	**Resource(s)**	**Dataset size**	**Evaluation measures**
	NB, DT, DL, kNN, RF, SVM, MLP, LSTM				
Punjabi (Singh & Kumar, 2019)	Supervised MLP, LSTM	Target WSD	Punjabi WordNet	66 words	MLP A:77.24% LSTM A:71.50%
Tamil (Sankaran & Vijay-Shanker, 2003)	Supervised Decision List	Target WSD	Morph analyser CIIL 3-million words corpus	4 words	P:90.72% R:72.19% F1:80%
Tamil (Baskaran & Vaidehi, 2004)	Unsupervised Clustering	Target WSD	CIIL 3-million words corpus	8 words	A:80.9%
Tamil (Rajendran, et al., 2013)	Unsupervised Clustering	Target WSD	Not Specified	6 words	Not Specified
Tamil (Anand Kumar, et al., 2014)	Supervised SVM	Target WSD	Not Specified	one word	Not Specified
Tamil (Padmamala, 2015)	Knowledge based Rule based approach	All-words WSD	WordNet, OneNet	3000 words (2000 (N), 1000 (V))	Not Specified
Tamil (Iswarya & Radha, 2017)	Unsupervised K-Means Clustering	Target WSD	Tamil POS tagger, Tamil and English WordNet, FIRE Tamil corpus	12 words	P:87.76% R:84.62% F:86.16% A:95.66%
Telgu (Palanati & Kolikipogu, 2013)	Unsupervised Decision List	All-words WSD	Telugu 100,000 words	Not Specified	A:97%
Telgu (Sreedhar, et al., 2014)	Knowledge Based Rule based	Not Specified	PoS rule dictionary	Not Specified	Set1 A:98% Set2 B:96%
Telgu (Durga Prasad, et al., 2018)	Supervised Statistical approach	Not Specified	Not Specified	Not Specified	C:100%
Telgu (Koppula, et al., 2019)	Knowledge Based Graph approach	Not Specified	Not Specified	Not Specified	Not Specified
Telgu (Durga Prasad, et al., 2019)	Supervised Conditional probabilities	Target WSD	30000 words	150 words	P:100% R:88% F:94%
Urdu (Naseer & Hussain, 2009)	Supervised Naïve bayes	Target WSD	18 million words	04 words	P: 98.35%, R:92.17%
Urdu (Arif, et al., 2016)	Supervised NB, SVM	Target WSD	Not specified	21 words	Not specified
Urdu (Abid, et al., 2018)	Supervised	Target WSD	2.5 million words	20 words	(NB) F:71% (DT) F:34.20%

Language Author(s)	Approach Classifier(s)	WSD Type	Resource(s)	Dataset size	Evaluation measures
	DT,SVM, Bayesian net				(SVM) F:34.30%
Urdu (Saeed, et al., 2019)	Supervised NB, SVM, kNN, ID3, MLP	Target WSD	222,533 token of tagged dataset 94.5 million tokens raw corpus	50 words	(NB-N) A:82.94%, (NB-V) A:77.06% (NB-ADJ) A:81.19%
Urdu (Saeed, et al., 2019)	Supervised	Target WSD	Indo WordNet, CLE Urdu WordNet, Urdu Lughat, 5,042 words tagged dataset	466 unique words	A: 57.71%

This section concludes with a description of the research studies extracted from different resources using the search strings. This comprehensive survey enables us to clarify all the research questions given in Table 2.1. In the following sub-sections, we have mapped the analysis outcomes with the research question framed.

2.4.3 Indian Languages Explored for the WSD

The answer to the RQ2 has been reported with the help of Figure 2.5 and 2.6. Figure 2.5 shows the research studies carried out in different Indian language families, and it clearly indicates that 74% of research work on WSD has been performed on Indo-Aryan languages, followed by 24% on Dravidian languages and 2% on Sino-Tibetan languages.

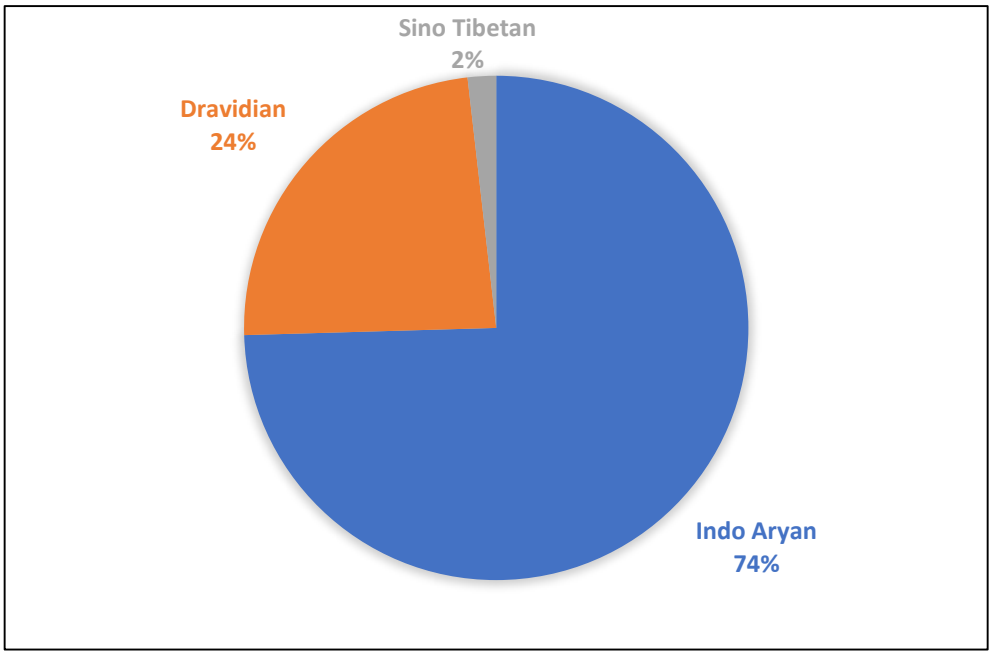


Figure 2.5: Status of the WSD research in different families of Indian languages

Figure 2.6 depicts the distribution of research studies among the different Indian languages. It has been observed from Figure 2.6 that the majority of research publications, 32%, are of Hindi language, followed by Bengali (13%), Punjabi (10%), Kannada (7%), Malayalam (6%), Assamese (5%), Tamil (5%), Telugu (5%), Urdu (5%), Guajarati (2%), Marathi (4%), Nepali (3%), Manipuri (2%).

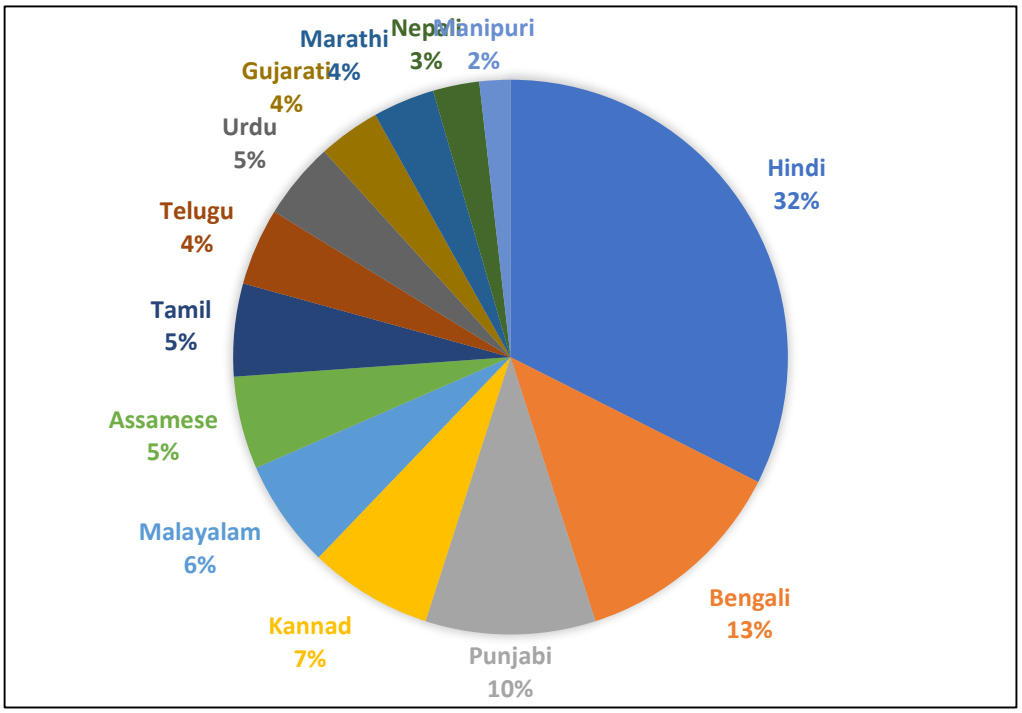


Figure 2.6: Status of the WSD research in different Indian languages

The analysis of the WSD system with respect to Indian languages shows that the most commonly investigated indian language for the development of WSD is the Hindi language, the official language of India. However, localization of computing, availability of the Unicode text, promotion of local languages by the Indian state governments has encouraged the researchers to perform the WSD for other Indian languages too. The availability of large text for Indian languages causes an urgent need to create a bridge between local language and Hindi or English languages. Because of this reason, the researchers are motivated to develop local language resources.

2.4.4 Resources for Implementation of the WSD

The knowledge base is the basic requirement for the implementation of the WSD. The knowledge resources are the sense inventories that represent the senses. There are two different categories of knowledge resources, structured and unstructured, that provide essential information to associate sense with words. The WSD task extracts the knowledge from the structured (Dictionaries, Thesauri, WordNet's and Ontologies) and unstructured resources

(Corpora and Word Wide Web text). To answer the RQ3, the analysis of knowledge resources used in different research studies of the WSD for Indian languages has been performed. The different types of knowledge bases developed and available for Indian languages are as follows.

2.4.4.1 Structured resources

WordNet, the most widely researched and developed structured resource, is a lexical database of words, and their respective meanings. WordNet groups the words into synonyms that represents a distinct concept. The quasi-synonyms, when read together, evoke a particular meaning to a human. Therefore, the architecture of WordNet is called cognitive architecture. The traditional lexical dictionaries arrange words according to the spellings, while WordNet organises words according to meanings. WordNet has five categories for lexicon words: nouns, verbs, adjectives, adverbs, and function words (Fellbaum, 1998). WordNet's words in these categories can be redundant, while this redundancy does not exist in traditional lexical dictionaries. However, this syntactically based division of words can be systematically exploited. WordNet establishes the similarity among the words with the help of the different semantic and lexical relations. The words are interconnected based on WordNet's different relations and constitute a substantial semantic network or directed acyclic graph (Fellbaum, 2017). This unique cognitive architecture of WordNet makes it more suitable for artificial intelligence-based NLP tasks than the traditional lexical dictionaries.

The first WordNet has developed by Princeton University and has been used as the de facto standard for English WSD (Navigli, 2009). Euro-WordNet has been developed from English WordNet by not only keeping all terms of synsets and the semantic relations for European languages (Dutch, Italian, Spanish, German, French, Czech and Estonian) but also extending and including the multilingual relations (Vossen, 2002). Balka-Net is an extension of Euro-WordNet to Balkan languages (Czech, Romanian, Greek, Turkish, Bulgarian, and Serbian) (Tufis, et al., 2004). The multi-WordNet is a multilingual database that contains aligned Italian WordNet with English WordNet (Pianta, et al., 2002). The multi-WordNet browser provides access to the Spanish, Portuguese, Hebrew, Romanian and Latin WordNet. The Global WordNet association is another effort that shares and links the WordNet's of worldwide languages and generates a multilingual global WordNet grid (Vintar & Fišer, 2016). The BabelNet is a significant unified lexical-semantic knowledge resource that is obtained by combining heterogeneous language sources such as WordNet, Wikipedia, Wikidata, Wiktionary and OmegaWiki (Navigli, et al., 2021). The BabelNet includes Wikipedia

information about the different Indian (as well as the Punjabi) languages, but it does not consist of the WordNet information of the different Indian languages.

IndoWordNet is a linked structure of WordNet having Indian languages namely Hindi, Marathi, Konkani, Nepali, Kashmiri, Assamese, Tamil, Malayalam, Telugu, Kannada, Manipuri, Bodo, Bangla, Punjabi and Guajarati (Bhattacharya, 2010). Hindi WordNet is the first Indian language WordNet started in the year 2000, and it has a maximum of 28,000 synsets (Kanojia, et al., 2018). Under the IndoWordNet project, the WordNet of Marathi, Konkani, Nepali, Kashmiri, Assamese, Tamil, Malayalam, Telugu, Kannada, Manipuri, Bodo, Bangla, Punjabi and Guajarati languages have been derived from the Hindi WordNet using an expansion approach. To generate the linked WordNet (an extension of WordNet), mapping of the Indo WordNets with the Princeton WordNet has been performed by (Kanojia, et al., 2018). WordNets for Indian languages are available at web link http://tdil-dc.in/indowordnet/.

IndoWordNet project is funded by the Department of Electronics Information Technology (DeitY), Ministry of Communication & Information Technology (MCIT), Govt. of India, and is executed by a consortium of nine academic institutions of India, namely the Goa University, Goa (the Consortium Leader); Indian Institute of Technology Bombay, Mumbai; Indian Statistical Institute, Kolkata; Dharmasinh Desai University, Nadiad; University of Kashmir, Srinagar; University of Hyderabad, Hyderabad; Punjabi University, Patiala; Thapar University, Patiala; and Jawaharlal Nehru University, New Delhi (Bhattacharyya, 2017). Hindi WordNet has made publicly available in 2006. The analysis of the research studies clearly indicates that the availability of Hindi WordNet enhances the research work on the Hindi WSD from the year 2009. The development of the Punjabi WordNet at Thapar University has motivated us to explore this lexical resource for the Punjabi WSD. The availability and accessibility of this resource encourage us to use it as a resource for the WSD. The Punjabi WordNet has been used as a sense inventory for the Punjabi WSD in this research work.

2.4.4.2 Unstructured Resources

These resources are a simple collection of text. The text can be raw or sense annotated. The raw corpus is the collection of text without any sense or word descriptions. The raw corpus is the collection of text from different online and offline resources. The PoS annotated corpus is the collection of words along with their standard PoS tags. The parallel corpus is the collection of texts in language L_1 and their translations into languages L_2. The comparable corpus is the collection of text from two or more languages that are similar in genre and topic without having

the same content. The different type of corpus (raw, comparable and PoS tag annotated) of the different Indian languages provided by different organisations are mentioned in Table 2.5. Recently, (Kakwani, et al., 2020) have developed monolingual corpora for 11 major Indian languages, sourced from news crawls. The authors have also developed pre-trained word embeddings, pre-trained language models, and multiple NLU evaluation datasets (IndicGLUE benchmark).

Table 2.5: Raw corpus available for Indian languages

Languages	**Type of corpus**	**Organisation/Online link**
22 official +YARAVA and KODAVA languages	Raw corpus	Linguistic Data Consortium for Indian Languages (LDC-IL) https://www.ldcil.org/resourcesTextCorp.aspx
Hindi, English, Punjabi, Urdu, Bangla, Oriya, Marathi, Gujarati, Konkani, Telugu, Tamil, Malayalam	Raw corpus	Indian Languages Corpora Initiative (ILCI) by Technology development by Indian language (TDIL) http://sanskrit.jnu.ac.in/ilci/index.jsp
Assamese, Bengali, Gujarati, Hindi, Kannada, Maithili, Malayalam, Marathi, Oriya, Punjabi, Tamil, Telugu	Raw corpus	The Central Institute of Indian Languages (CIIL) http://corpora.ciil.org/scheduled.html
Bengali, Gujarati, Hindi, Punjabi, Tamil, Urdu	Raw corpus	The Evaluations and Language resources Distribution Agency (ELDA) http://www.elra.info/en/about/elda/
English-Hindi	Parallel corpus	The IIT Bombay English-Hindi Parallel Corpus http://www.cfilt.iitb.ac.in/iitb_parallel/
English – Bengali, English – Dogri, English – Hindi, English – Kannada, English – Maithili and English - Nepali	Comparable corpus	Linguistic Data Consortium for Indian Languages (LDC-IL) https://www.ldcil.org/resourcesParallelTextCorp.aspx
Assamese, Bodo, Bengali, Gujarati, Hindi, Kannada, Kashmiri, Maithili, Malayalam, Manipuri, Oriya, Punjabi, Tamil and Urdu	PoS annotated corpus	Linguistic Data Consortium for Indian Languages (LDC-IL) https://www.ldcil.org/resourcesSampleTextCorp.aspx
Assamese, Bengali, English, Gujarati, Hindi, Kannada, Malayalam, Marathi, Oriya, Punjabi, Tamil, Telugu	Raw Corpus	https://indicnlp.ai4bharat.org/corpora/

The raw corpus listed in Table 2.5 has been extensively explored by the researchers for the WSD tasks. The raw corpus of the Bangla language prepared by ILCI by TDIL has been explored in the supervised NB approach of the Bangla WSD (Pal, et al., 2015a). A Knowledge base approach of WSD for the Bengali language has explored the PoS annotated corpus of Bangla language by ILCI by TDIL (Pal, et al., 2017b). Unsupervised sense induction and Maximum Entropy approaches of Bangla WSD have also explored the raw corpus of Bangla language prepared by ILCI by TDIL (Pal & Saha, 2019). A knowledge base approach of Hindi WSD has experimented on the Hindi raw corpus by TDIL (Athaiya, et al., 2018). Tamil raw corpus by CIIL has been explored for the Tamil WSD (Baskaran & Vaidehi, 2004). The raw corpus of Punjabi language by ELDA has been leveraged in the supervised Punjabi WSD (Walia, et al., 2017; Walia, et al., 2019).

Further, the sense tagged or sense annotated corpus contains the senses of the words in the context. (Chatterjee, et al., 2010) have developed a Graphical User Interface (GUI) tool for Indian languages called Sense Marking Tool, which marks each word of the context with its sense selected from their respective WordNet. This sense marker tool supports nine languages (English, Hindi, Marathi, Tamil, Telugu, Kannada, Malayalam, Bengali and Punjabi) and adopts the sequential tagging approach in which each word is tagged in the sequence as it occurs in the context. (Khapra, et al., 2010) have developed and exploited the sense tagged datasets of the English, Hindi and Marathi languages to evaluate the supervised approaches for WSD. The knowledge base and supervised approaches for the Hindi WSD have experimented on sense tagged 60 polysemous Hindi nouns having 7506 instances (Singh & Siddiqui, 2015). Table 2.6 shows the sense annotated corpus of the different Indian languages. The evidence on the structured (IndoWordNet) and the unstructured (raw corpus and sense annotated) resources given in Table 2.5, and 2.6 helps us in attaining the answer to RQ3.

Table 2.6:Sense annotated corpus for Indian languages

Language	**Domain /Type**	**Organisation/Online link**
Hindi, Marathi, English	Tourism, Health	IIT Bombay http://www.cfilt.iitb.ac.in/wsd/annotated_corpus/
Hindi	60 polysemous nouns	TDIL https://www.tdil-dc.in/

It has been observed from analysis of research studies that the emergence of the WordNet for various Indian languages based on the IndoWordNet has encouraged the researchers to

experiment with different knowledge base algorithms such as Lesk overlap, selectional preferences, association rules and graph-based approaches for the Indian language WSD.

Figure 2.7 depicts the utilisation of WordNet for implementing the WSD of different Indian languages. It has been observed from Figure.2.7 that WordNet is the critical resource for the WSD and has primarily applied by Indo-Aryan languages.

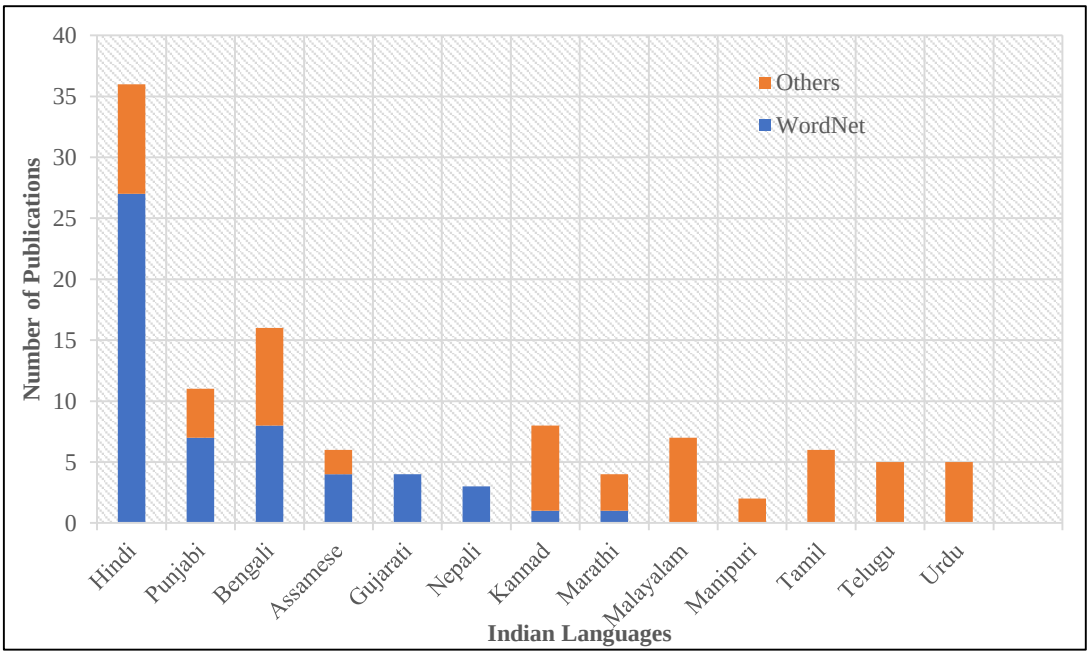


Figure 2.7: Use of WordNet in the WSD approaches for Indian languages

2.4.5 Context Processing Resources

The context processing resources such as the shallow parser, morph analyser, PoS tagger and dependency parser transforms the context of the knowledge base resources into a piece of suitable structured information. The output of the context processing tools is a structured text that has been used as an input to an automatic WSD technique. To address the research question RQ4, the research studies have been analysed to determine the different context processing resources. Table 2.7 summarizes the list of the online available language pre-processing and context processing tools developed for different Indian languages. This list of available context processing tools delivers the RQ4 answer.

Table 2.7: Tools for Indian languages

Sr.No.	Resource	Language	Link
1	Shallow parser	Hindi, Punjabi, Urdu, Tamil, Bengali, Telugu, Kannada, Malayalam and Marathi	http://ltrc.iiit.ac.in/showfile.php?filename=downloads/shallow_parser.php
2	Morph Analyser	Hindi, Marathi, Kannada and Telugu	http://ltrc.iiit.ac.in/showfile.php?filename=onlineServices/morph/index.htm
3	PoS Tagger	Hindi, Kannada and Telugu	http://sivareddy.in/downloads
		Punjabi	http://punjabi.aglsoft.com/punjabi/?show=tagger http://punjabipos.learnpunjabi.org/
4	Dependency Parser	Hindi	https://bitbucket.org/sivareddyg/hindi-dependency-parser
5	Lemmatizer	Hindi, Assamese, Bengali, Bodo, Gujarati, Kannada, Kashmiri, Konkani, Malayalam, Manipuri, Marathi, Nepali, Sanskrit, Tamil, Telugu, Punjabi, Urdu and Odiya	http://www.tdil-dc.in/index.php?option=com_download&task=showresourceDetails&toolid=1098&lang=en

2.4.6 Approaches for the WSD

The analysis of research studies states that approaches explored for the WSD of Indian languages can be broadly classified into supervised, unsupervised, knowledge based approaches. Knowledge based, Unsupervised, and Supervised approaches. These approaches have distinguished based on the amount of supervision and knowledge used for sense

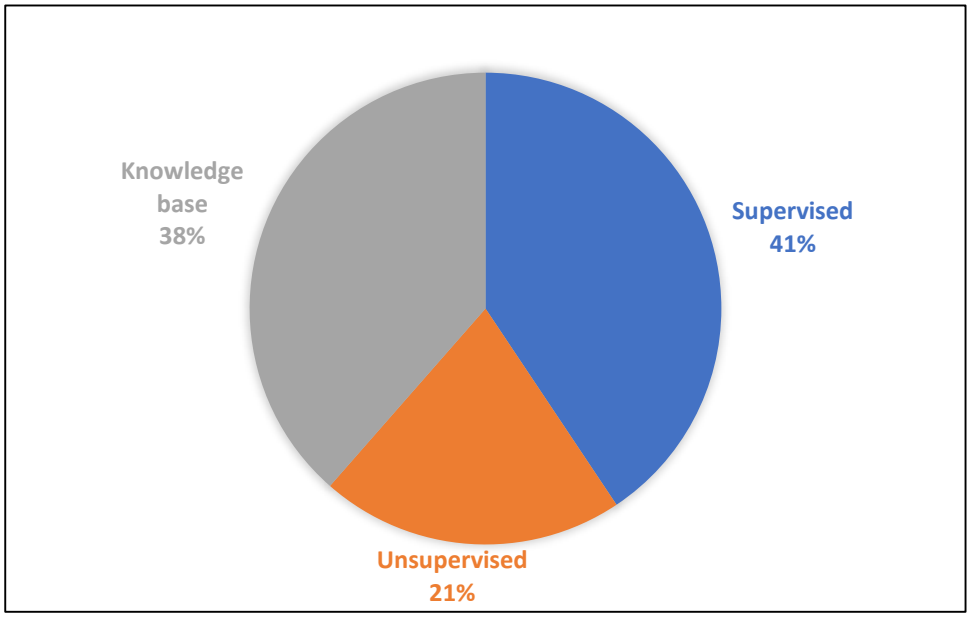


Figure 2.8: WSD approaches applied for Indian languages

disambiguation. Figure2.8 depicts the percentage number of studies belong to the supervised, unsupervised, knowledge based approaches. Supervised approaches are the most successful approaches in the field of WSD. Supervised approaches have been applied by the researchers on 41% of Indian languages for sense disambiguation. 38% of the research studies on the WSD for Indian languages have applied the knowledge base approaches. These approaches explore the lexical computational resource such as the WordNet to ascertain the sense of the ambiguous word. The 21% of research studies have applied unsupervised approaches and have explored the raw corpus of Indian languages to accomplish the WSD. It has been observed from Figure2.8 that supervised approaches are the most commonly exploited WSD classification techinques for Indian languages.

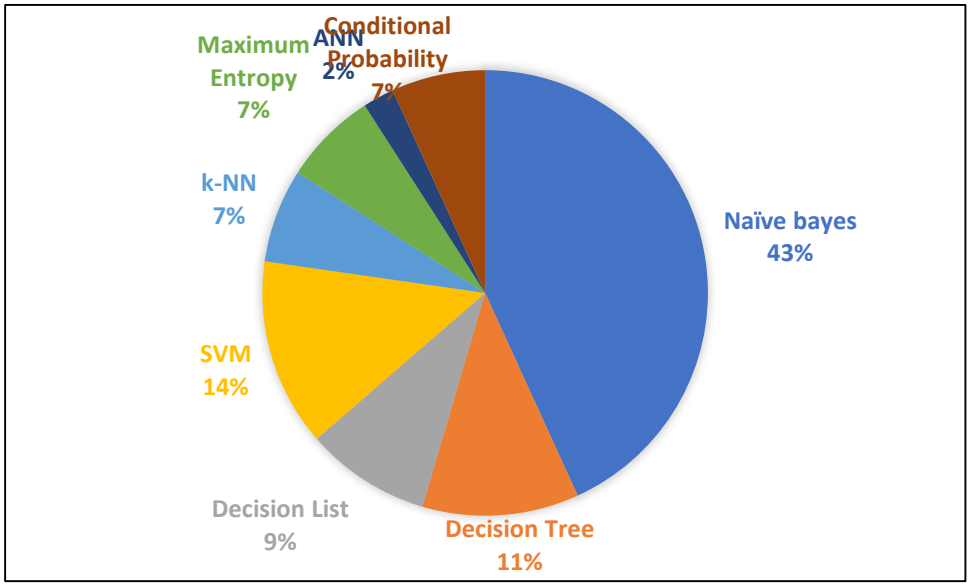


Figure 2.9: Supervised WSD approaches applied for Indian languages

Figure 2.9 shows the percentage number of different supervised ML approaches of WSD investigated for the Indian languages. The most prominent supervised approach explored for the WSD of Indian languages is the NB (43%), followed by the SVM (14%) and DT (11%). Thus, the NB, SVM, and DT approaches constitute nearly 70% of the research studies of the WSD systems for Indian languages.

The analysis of the research work of WSD for Indian languages segregates WSD systems into target and all-words types. Figure 2.10 depicts the percentage of two different types of WSD systems (target and all-words) applied for Indian languages. The reason behind the development of the target WSD system for Indian languages is the availability of WordNet and

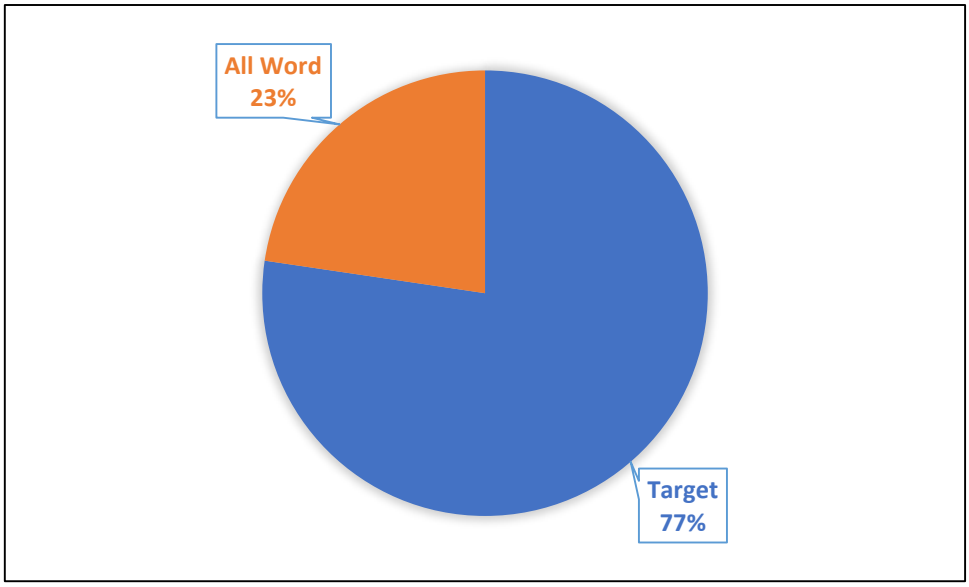


Figure 2.10: Types of the WSD system for Indian languages

ease of evaluation. The limited research work has been reported for all-words WSD due to the non-availability of the large sense tagged datasets for Indian languages.

It has been observed from Figure 2.10 that the targeted WSD is the most common type of WSD used by researchers for the Indian language WSD. Table 2.8 reports the supervised techniques and the count of research studies of different Indian languages that have applied them to achieve the WSD.

Table 2.8: Supervised WSD approaches applied for Indian languages

Techniques	**Languages (Count)**
NB	Assamese (4), Bengali (4) ,Hindi(3), Punjabi (4), Urdu(3), Kannada(2), Malayalam (1)
DT	Assamese (2), Hindi(1), Manipuri(1), Urdu(1), Punjabi (1)
DL	Hindi(1), Kannada(1)
SVM	Hindi(2), Malayalam(1), Manipuri(1) ,Tamil(1) ,Urdu(1)
k-NN	Bengali (1), Punjabi (1)
Maximum Entropy	Marathi (1)
ANN	Bengali (1), Hindi(1)
Conditional Probability	Hindi(1)
Semantic Similarity	Hindi(2), Urdu(1)
Deep learning	Gujarati(1), Punjabi (2)

The research question RQ5 has been accomplished through Figure 2.8, 2.9, 2.10 and Table 2.8, which summarizes that the supervised approaches for WSD have been used by researchers for most Indian languages. From Table 2.8, it can be stated that the NB classifier for WSD has experimented over almost all Indian languages. It has been further observed that there is limited application of deep learning approaches for the WSD of Indian languages.

2.4.7 Evaluation Metrics for the WSD

Researchers have investigated the WSD using different knowledge resources, sense inventories and test datasets. Thus the comparison and evaluation of the different WSD approaches become highly complex. Nevertheless, the evaluation measures for the independent WSD system have been borrowed from the information retrieval fields such as accuracy, precision and F1 measure (Navigli, 2009). Accuracy (A) is the most commonly used evaluation measure. The researchers have used accuracy evaluation measures when the WSD system assigns senses to all the test instances of the dataset. However, the metrics Precision, Recall and F_1 have also been applied by the researcher when the WSD systems do not assign senses to all the test instances of the dataset. The metrics such as precision, recall, coverage and F1-score used for the evaluation of the WSD system are as follows.

- Precision is the percentage of correct answers determined by the automatic system to the number of answers given by the system, as shown in equation (2.2).

$$Precision = \frac{\text{Number of correct answers}}{Number\ of\ answers\ to\ be\ given} \quad \ldots (2.2)$$

- Recall is defined as the ratio of the number of correct answers given by the automatic system to the total number of answers to be given, as shown in equation (2.3).

$$Recall = \frac{\text{Number of correct } answers}{Total\ number\ of\ answers\ to\ be\ given} \quad \ldots (2.3)$$

  Generally, the Recall value is always less than the Precision value. However, when the Coverage of the automatic system is 100%, that is, the "*number of answers provided*" is equal to the "*total number of answers to provide",* then the Recall value is equal to the Precision value. In the WSD literature, recall is also denoted as accuracy.

- The F1 measure, also known as F-score, determines the weighted harmonic mean of recall and precision. Equation (2.4) defines the *F1-measure*.

$$F_1 = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad \ldots (2.4)$$

  The F1 measure is used for the validation of results where the coverage of the WSD systems is less than 100%.

It has been observed from the analysis of the research studies that the precision, recall and F1 metrics are used for the evaluation of the WSD for Indian languages and thus answers the RQ6.

To standardize the evaluation process for WSD, it requires the standard datasets and parameters. Senseval (now renamed Semeval) is the series of evaluation workshops that have performed a comparative evaluation of WSD systems based on languages, tasks, lexical substitution, semantic role labelling. Table 2.9 mentions the details of these competitions.

Table 2.9: Senseval/Semeval workshop details

Competition	**Languages**	**Resource(s)**	**WSD task(s)**	**Best performing approach(s)**
Senseval-1	English, French, and Italian	HECTOR	Lexical sample task	Decision List
Senseval-2	Czech, Dutch, English, Estonian, Basque, Chinese, Danish, English, Italian, Japanese, Korean, Spanish, Swedish	WordNet 1.7	All-words and lexical sample	Supervised systems outperformed unsupervised approaches
Senseval-3	Basque, Catalan, Chinese, English, Italian, Romanian, Spanish	WordNet 1.7.1	All-words, lexical sample, gloss disambiguation, semantic role labeling, multilingual annotations, logic forms, and the acquisition of subcategorizations	Top systems used kernel methods. Others applied voted combination of algorithms rich set of features, comprising domain information and syntactic relations
Semeval-2007	Arabic, Catalan, Chinese, English, Spanish, Turkish	coarse-grained inventories	All-words, lexical sample, lexical substitution task, word sense induction, and discrimination along with semantic analytic task	Coarse-grained supervised systems outperformed the fine grained system.
SemEval2010	Catalan, Chinese, Dutch, English, French, German, Italian, Japanese, Spanish	WordNet, Euro-parl parallel corpus, Japanese corpus	All-words, lexical sample, Word sense induction, Domain specific WSD	Word sense induction, domain specific all word disambiguation has performed.
SemEval2012	Chinese, English	Not mentioned	Word Similarity	Similarity based tasks have performed
SemEval2013	Catalan, French, German, English, Italian, Spanish	WordNet 3.0, HowNet and	All-words, lexical sample, Word sense	Word sense induction and Chinese lexical sample as well as all-words task

Competition	Languages	Resource(s)	WSD task(s)	Best performing approach(s)
		Chinese adverb corpus	induction, Domain specific WSD	
SemEval2014	English, Spanish, French, German, Dutch	English Spanish parallel corpus	Similarity and Sentiment analysis	WSD used for textual similarity.
SemEval2015	English, Spanish, Arabic, Italian	BabelNet 2.5,WordNet 3.0, OmegaWiki, Open Multilingual WordNet synsets	Word Sense Disambiguation and Induction	Multilingual all word WSD, entity linking, analysis of clinical text and corpus pattern analysis have performed

The analysis of research studies of the WSD for Indian languages indicates the nonavailability of a standard dataset for the Indian languages. However, the preparation of such standard datasets for Indian languages has been initiated. Table 2.7 shows the available standard sense tagged datasets for Hindi and Marathi languages. Recently (Saeed, et al., 2019) have prepared a standard all word Urdu corpus using the Urdu Lughat dictionary.

2.5 Findings of the Systematic Review

In this section, the outcomes of the analysis of research studies have discussed by answering all the research questions in Table 2.1. To address RQ7, the analysis of research studies of the WSD for Indian languages has been performed to determine the availability of any online/offline WSD system for Indian languages. It has been observed that currently, there does not exist any independent WSD system that can perform the WSD for Indian languages.

The significant outcomes of the research questions given in Table 2.1 from this methodical and comprehensive survey on the WSD for Indian languages are as follows.

- The WSD for Indian languages started in 2002 with the development of the Hindi WordNet. The first work in the field of the WSD for the Indian language reported in 2002 by (Unnikrishnan & P, 2002) for the Hindi language. However, the availability of the Hindi WordNet and the development of the IndoWordNet has given a major thrust to the research in the WSD for Indian languages since 2009.
- The analysis of research studies states that most of the research in the WSD for the Indian languages field has been performed after 2009. Indo-Aryan family of Indian languages have mostly explored for the WSD task, and the Hindi language is the primary language of research among the Indo-Aryan languages, followed by Bengali and Punjabi languages.

- The implementation of the IndoWordNet, a computational lexical resource for 18 scheduled languages of India, has initiated the research work for Assamese, Bengali, Hindi, Kannada, Manipuri, Marathi, Malayalam, Nepali, Punjabi, Tamil, Telugu, Urdu languages.
- This research study gives the description, availability and accessibility of the knowledge resources such as WordNet, annotated dataset and raw corpus for Indian languages.
- This systematic review provides information about the context processing tools such as Morph analyser, POS tagger, and dependency parser employed for processing the context of different Indian languages.
- The literature survey gives an insight into the WSD (targeted, all-words) types and approaches (supervised, unsupervised and knowledge based) implemented for Indian languages.
- The research studies for the WSD of Indian languages show that the traditional ML techniques are the most generally applied approaches. Nonetheless, the emerging deep learning techniques are also exciting the researchers due to their improved accuracy and domain independence. The NB classifier is the most explored classifier for Indian languages.
- The researchers have primarily experimented with the traditional ML techniques on target-based WSD using WordNet as a lexical resource.

Consequently, this methodical literature review provides the details of the WSD approaches, raw corpus, sense tagged corpus, online context processing tools and evaluation parameters applied for different Indian languages. The researchers can further investigate these resources to implement sense disambiguation in different languages and other NLP tasks.

Chapter Summary

WSD is crucial for multilingual NLP tasks. The evolution of the WSD has been discussed in this chapter. India is a multilingual country that has 22 official languages that make the WSD relevant in the Indian context. The increase in the experimentation of the WSD for the Indian language has motivated us to conduct this systematic survey. The availability of the datasets and knowledge bases for the Indian languages have encouraged us to explore the WSD for Indian languages. The research work on the WSD in context to Indian languages has commenced with the development of the Hindi WordNet by (Narayan, et al., 2002). Afterwards the expansion of the Hindi WordNet to other Indian languages has provided an impetus in the research of the WSD for different Indian languages. Presently, there exists no research study that covers a thorough examination of Indian languages in the domain of WSD. Hence, this chapter contributes to the literature on the WSD for Indian languages. This literature survey includes the methodical review of 110 research studies of WSD for Indian languages published from 2010 to 2020.

The systematic research of the WSD for Indian languages has been performed according to the sketched research questions. The answers to these outlined questions have been attained by examining the shortlisted 110 research studies selected from the different e-databases. The inclusion and exclusion conditions have delineated and followed in selecting the most relevant research studies of the WSD for Indian languages.

A summary of the WSD types, approaches and computational linguistic sources for the different Indian language is provided in this chapter. This survey shows that most WSD work has been performed in the Hindi language, followed by the Bengali and the Punjabi languages of the Indo-Aryan family. It has been analysed that the researchers have mainly used ML (i.e., 40%) approaches for the WSD for Indian languages in comparison to other unsupervised and knowledge based approaches. The implementation and accessibility of computational resources like WordNet have significantly improved the research of the WSD for Indian languages. The details of the available raw corpus and sense tagged corpus of different Indian languages have been reported in this chapter. The context processing resources such as shallow parsers, lemmatizer, dependency parser, morph analyser have also been given in this chapter. The researchers have performed the WSD work at a targeted word and have mostly explored ML classifiers.

The summary about the different types of the WSD approaches, lexical resources, and evaluation measures indicates that research is more focused on the Hindi language. There is an urgent need to build the resources and to do research on the WSD for other Indian languages. The IndoWordNet is such an effort to create a standard lexical resource for 18 Indian languages. The availability of the Punjabi WordNet as a part of the IndoWordNet has motivated us to perform the Punjabi WSD. The non-availability of a sense annotated dataset and standard online WSD system for the Punjabi language has encouraged us to develop and implement an independent Punjabi WSD. It has been observed from this survey that the research for the Punjabi WordNet, Punjabi sense tagged dataset, and Punjabi WSD has been emerging.

Chapter 3 : Data Collection and Corpus Creation

In this chapter, the creation of sense tagged corpus for the ambiguous Punjabi noun words has been discussed. The creation of sense tagged corpus has divided into three subtasks, namely the selection of ambiguous words, merging of senses and the sense tagging process. In selecting ambiguous words subsection, the Punjabi WordNet, a Punjabi language knowledge resource, has been elaborated. The Punjabi ambiguous noun words have been selected from the Punjabi WordNet. The Punjabi WordNet defines the meaning generally called senses of Punjabi words. However, the senses of Punjabi ambiguous noun words selected from the Punjabi WordNet are overlapping senses that are they have related meanings. The sense merging process has presented to produce the discerning coarse senses of ambiguous words. The objective of sense merging is to define the distinguishing senses of the Punjabi noun words. The sense tagging process subsection presents the detailed process of gathering the Punjabi language text for the different senses of the Punjabi ambiguous noun words. The details of the sense tagged dataset and inter-annotator agreement is presented in the sense tagging process section. The preparation of the sense tagged dataset aims to implement the supervised WSD for the Punjabi language.

3.1 Punjabi Sense Tagged Corpora Creation Process

The preparation of sense tagged copra for a language is domain-specific, time-consuming and needs considerable human efforts. In this research work, the Punjabi language sense tagged dataset has been prepared manually. The creation of the sense tagged corpora for the Punjabi language has been classified into three steps, as shown in Figure 3.1. In the first step, the Punjabi noun words having multiple meanings (polysemous) are identified. The polysemous Punjabi noun words have been selected from the knowledge resource Punjabi WordNet. The Punjabi WordNet a lexical knowledge resource has analysed to collect additional information such as glosses, examples and relational information about the ambiguous Punjabi noun words. The senses of the Punjabi noun words defined in the Punjabi WordNet are very fine and overlapping. The clear distinction among senses of the words is essential for a classification task like WSD (Ide & Wilks, 2007).

The second step delineates the coarse senses of the selected ambiguous Punjabi nouns from Punjabi WordNet with the help of the sense merging algorithm. In the third and the final step,

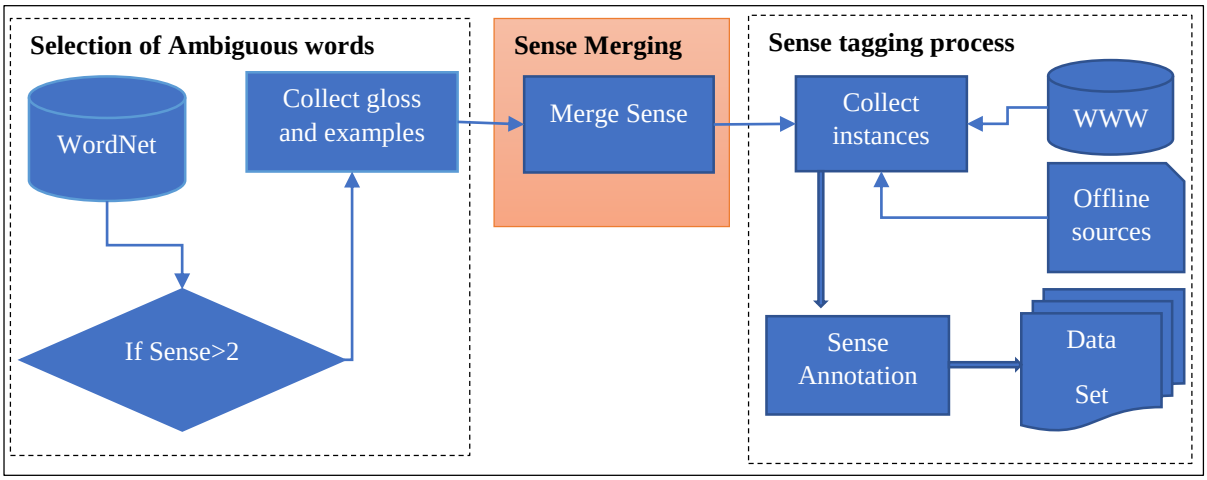


Figure 3.1: Sense tagged dataset preparation process

the context for different coarse senses of the Punjabi language ambiguous words has been collected. The resources such as the world wide web, online news, blogs, books and offline books have been searched to collect the instances for different senses of the ambiguous noun words. Three different language experts have validated the collected instances of different Punjabi noun words. The sense tagged dataset of an individual Punjabi noun word has collected in the different text files. These sense files constitute the sense tagged dataset for the ambiguous Punjabi noun words. The subtle details of the individual steps of the sense tagged dataset preparation process are discussed in the following subsections.

3.2 Selection of the Punjabi Ambiguous Words

An ambiguous word in a given natural language has multiple interpretations based on the context in which it has been used. For example, the Punjabi polysemous noun word ਉੱਤਰ (TT: uttar) has two different meanings Sense1 (Answer) and Sense2 (North) in two different sentences as shown in Table 3.1.

Table 3.1: Different meaning of the Punjabi ambiguous word ਉੱਤਰ (TT: uttar)

Ambiguous word	**Senses**	**Example instances**
"ਉੱਤਰ" {(TT): uttar, Part of Speech (PoS): Noun}	Sense1 (**Answer**)	P: ਤੁਸੀ ਮੇਰੇ ਪ੍ਰਸ਼ਨ ਦਾ ਉੱਤਰ ਨਹੀਂ ਦਿੱਤਾ {Transliteration (TT): tusī mērē prashan dā uttar nahīṃ dittā} {Translation (T): You have not **answered** my question}
	Sense2 (**North**)	P: ਭਾਰਤ ਦੇ ਉੱਤਰ ਵਿਚ ਹਿਮਾਲਿਆ ਪਰਬਤ ਵਿਰਾਜਮਾਨ ਹੈ {TT: bhārat dē uttar vic himāliā parbat virājmān hai} {T: The Himalayas is in **north** of India}

The words that are represented by multiple meanings are known as polysemous words. The words that have a single meaning are known as monosemous words. The definitions of different meanings of the words based on their Part of Speech (PoS) information have been represented in the sense inventories such as machine readable dictionaries, Thesauri, Ontologies and WordNet. The extensively explored sense inventory for different NLP tasks of different languages is the WordNet. The Punjabi WordNet has been used to prepare the sense tagged dataset of Punjabi language by selecting the polysemous Punjabi words and their senses from it. The detailed attributes and design of the Punjabi WordNet are discussed in the following subsection.

3.2.1 Punjabi WordNet

The Punjabi WordNet is one of the constituents of the IndoWordNet, which is a linked structure of WordNet's of major Indian languages (Bhattacharyya, 2017). The Punjabi WordNet has been developed by Thapar University and Punjabi University of Patiala, Punjab, India. The availability and accessibility of the Punjabi WordNet have motivated us to use it as a resource for the Punjabi WSD.

The expansion approach has applied to develop the Indian languages WordNet's from the Hindi WordNet. Because the Hindi language has relations with almost all the ancient Indian languages, and it has acceptability in pan India. In the expansion approach, synset's of different Indian language WordNet's have created from the Hindi WordNet. The structured Punjabi WordNet has derived and developed from the Hindi WordNet using the expansion approach. The building of WordNet is based on principles of minimalist (minimum words to represent a concept), coverage (completion of synset) and replaceability (all the words in a synset should be able to replace one another in an example sentence) (Bhattacharya, 2010).

3.2.1.1 The Punjabi WordNet Structure

The structure of the Punjabi WordNet consists of synsets and relationships among them. The synsets are defined as the collection of the synonym Punjabi words. One or more synonymous words have elaborated the synset of a word. The senses of a word have distinguished according to its PoS categories, *viz.*, nouns, verbs, adjectives and adverbs. The Punjabi WordNet defines each synset with gloss and the respective example sentences. The gloss of a word hints at the meaning of a sense and is defined for each synset. The gloss of a word has further been elaborated by using the example instances. For illustration, the synonyms, gloss and example instances for an ambiguous Punjabi noun word 'ਜਰ' (TT: jar) are shown in Table 3.2.

Table 3.2: Gloss and examples of different senses ids of word ਜਰ (TT: jar) selected from the Punjabi WordNet

ਜਰ$_n$ Syn_ID{446}	
Synonyms	ਤਾਕਤ (TT: tākat), ਸਮਰੱਥਾ (TT: samratthā), ਸ਼ਕਤੀ (TT: shaktī), ਬਲ (TT: bal), ਜਰ (TT: jar), ਜ਼ੋਰ (TT: zōr)
Gloss	P: ਸਮਰੱਥਾ ਨਾਲ ਪੂਰਨ ਹੋਣ ਦੀ ਅਵਸਥਾ ਜਾਂ ਭਾਵ (TT: samratthā nāl pūran hōṇ dī avsathā jāṃ bhāv)
Example	P: ਤੁਹਾਡੀ ਤਾਕਤ ਦੇ ਕਾਰਣ ਹੀ ਇਹ ਕੰਮ ਹੋ ਸਕਿਆ ਹੈ (TT: tuhāḍī tākat dē kāraṇ hī ih kamm hō sakiā hai)
Gloss in Hindi	क्षमता से पूर्ण होने की अवस्था या भाव:"आपकी ताक़त के कारण ही यह कार्य हो सका (TT: kshamata se poorn hone kee avastha ya bhaav:"aapakee taaqat ke kaaran hee yah kaary ho saka)
Gloss in English	enduring strength and energy
ਜਰ$_n$ Syn_ID{3051}	
Synonyms	ਧਨ-ਦੌਲਤ (TT: dhan-daulat), ਧਨ-ਦੋਲਤ (TT: dhan-dōlat), ਦੌਲਤ (TT: daulat), ਰੁਪਏ (TT: rupaē), ਰੁਪਏ-ਪੈਸੇ (TT: rupaē-paisē), ਪੈਸੇ (TT: paisē), ਪੈਸਾ-ਧੇਲਾ (TT: paisā-dhēlā), ਰੁਪੋੜ (TT: rupōṛ), ਜਰ (TT: jar), ਲਕਸ਼ਮੀ (TT: lakshamī)
Concept	ਰੁਪਏ (TT: rupaē), ਪੈਸੇ (TT: paisē), ਸੋਨਾ-ਚਾਂਦੀ (TT: sōnā-cāndī), ਜਮੀਨ-ਜਾਇਦਾਦ (TT: jamīn-jāidād) ਆਦਿ (TT: ādi)
Example	P: ਜਰ ਦਾ ਉਪਯੋਗ ਚੰਗੇ ਕੰਮਾਂ ਵਿਚ ਹੀ ਕਰਨਾ ਚਾਹੀਦਾ ਹੈ (TT: jar dā upyōg caṅgē kammāṃ vic hī karnā cāhīdā hai)
Gloss in Hindi	सोना-चाँदी, ज़मीन-जायदाद आदि संम्पत्ति जिसकी गिनती पैसे के रूप में होती है :"धन-दौलत का उपयोग अच्छे कार्यों में ही करना चाहिए (TT: sona-chaandee, zameen-jaayadaad aadi sammpatti jisakee ginatee paise ke roop mein hotee hai :"dhan-daulat ka upayog achchhe kaaryon mein hee karana chaahie)
Gloss in English	wealth reckoned in terms of money; "all his money is in real estate"
ਜਰ$_n$ Syn_ID{5441}	
Synonyms	ਜੰਗ (TT: jaṅg), ਜਰ(TT: jar)
Concept	P: ਲੋਹੇ ਆਦਿ ਤੇ ਚੜਨ ਵਾਲਾ ਉਹ ਕਾਲਾ ਅੰਸ਼ ਜੋ ਹਵਾ ਅਤੇ ਨਮੀ ਦੇ ਪ੍ਰਭਾਵ ਤੋ ਉਤਪੰਨ ਹੁੰਦਾ ਹੈ (TT: lōhē ādi tē caṛan vālā uh kālā aṃsh jō havā atē namī dē prabhāv tō utpann hundā hai)
Example	P: ਲੋਹੇ ਨੂੰ ਜਰ ਲੱਗ ਗਿਆ ਹੈ (TT: lōhē nūṃ jar lagg giā hai)
Gloss in Hindi	लोहे आदि पर चढ़ने वाला वह काला अंश जो हवा और नमी के प्रभाव से उत्पन्न होता है:"लोहे पर जंग लग गया है (TT: lohe aadi par chadhane vaala vah kaala ansh jo hava aur namee ke prabhaav se utpann hota hai:"lohe par jang lag gaya hai)
Gloss in English	a red or brown oxide coating on iron or steel caused by the action of oxygen and moisture

Besides describing the sense of the Punjabi ambiguous noun in Punjabi gloss, the gloss in Hindi and English language have also given for it. The google translate tool has produced the Hindi to English transliteration shown in Table 3.2. The Punjabi *Gtrans* tool has applied to generate Punjabi to English transliterations.

The synset's defined in the Punjabi WordNet are interlinked, employing semantic and lexical relations (Kaur, et al., 2010). Lexical relations define a relationship between two words directly within the Punjabi WordNet. On the other hand, semantic relations defined in the WordNet, describe relationships between synset of the Punjabi WordNet. The different lexical and semantic relations described in the Punjabi WordNet are given in the following subsections.

3.2.1.2 Lexical Relations

The different lexical relations that connect word senses included in different synset's are as follows.

- **Antonymy:** This relation expresses the contrasting relation among the concepts. This lexical relation holds for all grammatical classes of the words. This relation is defined by the words present in the two different concepts. Antonymy relation is complex, symmetric and is not a semantic relation. For example, the word ਸੁੱਖ (TT: sukkh) (T: pleasure) has antonym word ਦੁੱਖ (TT: dukkh) (T: suffering).
- **Gradation:** This lexical relation represents an intermediate concept between the two opposite concepts. For instance, words ਬਚਪਨ (TT: bacpan) (T: childhood) and ਬੁੱਢਪਾ (TT: buḍḍhpā) (T: old age) are opposites to each other and ਜਵਾਨੀ (TT: javānī) (T: youth) is the intermediate concept.
- **Derived form:** This lexical relation explores the root form derived from a particular word. The derived form normalizes the relations from noun to verb, noun to an adjective or vice versa. For instance, the word ਕ੍ਰਮਵਾਰ (TT: kramvār) (T: step by step) derived from the words ਕ੍ਰਮ (TT: kram) (T: series).

3.2.1.3 Semantic Relations

The semantic relations are defined among the complete synset. The semantic relations, namely hypernymy, hyponymy, meronymy, holonymy, entailment, causation, troponymy, ability link, capability link, functional link, attributes, modifies noun and modifies verb are as follows.

- **Synonymy:** The similarity between the two concepts is called the synonymy relation. There is symmetric relation between two synonymy concepts, that is if '*c1*' is similar to '*c2*' then '*c2*' is also similar to '*c1*'. The synonymy or the synset for the word ਚੀਰ (TT: cīr) is {ਮਾਂਗ (TT: māṅg), ਚੀਰ (TT: cīr)}.

- **Hypernymy/Hyponymy:** The hypernymy relation is also called '*kind of*' or '*is a*' relation. The hypernymy relation captures the superset hood. The hyponymy relation is the inverse of hypernymy. It captures the sub-set relations. The hyponymy relation is transitive and asymmetrical. The synset {ਮਾਪਕ (TT: māpak), ਮਾਪਣ (TT: māpaṇ), ਪੈਮਾਣਾ (TT: paimāṇā), ਨਾਪ (TT: nāp), ਉਪਕਰਣ (TT: upkaraṇ), ਮਾਪ ਕਰਨ ਵਾਲਾ ਯੰਤਰ (TT: māp karan vālā yantar)} (T: measurement) is a superset of the ਗਜ਼ (TT: Gaz) (T: a specific unit of measurement) and, thus it describes the hypernym relation. Whereas the word ਗਜ਼ (TT: Gaz) is a kind of ਨਾਪ (TT: nāp). Thus the word ਗਜ਼ (TT: Gaz) defines hyponym relation with the ਨਾਪ (TT: nāp). The hypernym/hyponymy generates a hierarchical semantic structure, in which a hyponym is said to be below its hypernym. These hierarchical representations have been extensively explored in the information retrieval systems (Kaur, et al., 2010).
- **Meronymy/Holonymy:** The meronymy/holonymy relations encode a '*part-of*' relation and its inverse. It is defined as if the concept '*x*' is the part of concept '*y*', then '*x*' is meronymy of '*y*' and '*y*' is holonymy of '*x*'. For example, the words ਜੜ੍ਹ (TT: jaṛh) (T: root) and ਪਤੇ (TT: patē) (T: leaves) are the meronymy of the word ਦਰਖਤ (TT: darkhat) (T: tree) while ਦਰਖਤ (TT: darkhat) is a holonym of the words ਜੜ੍ਹ (TT: jaṛh) and ਪਤੇ (TT: patē). The meronymy relation is transitive and asymmetrical. The meronymy relations represent associations in the form of a network. The position of a word in this relationship network helps in determining its sense.
- **Entailment:** The verb '*Y*' is entailed by a verb '*X*' if by doing '*X*' you must be doing '*Y*'. This relationship is unidirectional. For instance, the verb synset ਭੱਜਣਾ (TT: bhajjṇā) (T: running) has entailment relation with the verb ਕਦਮ ਚੁੱਕਣਾ (TT: kadam cukkṇā) (T: taking a step). The relation is unidirectional as ਕਦਮ ਚੁੱਕਣਾ (TT: kadam cukkṇā) (T: running) does not entail ਭੱਜਣਾ (TT: bhajjṇā) (TT: bhajjṇā).
- **Troponymy:** Troponymy relation describes the elaboration between two verbs. It shows a manner of action. Troponym term has used for the verbs instead of a hyponym. For instance, the verb ਮੁਸਕਰਾਉਣਾ (TT: muskarāuṇā) (T: smiling) is a troponym of the verb ਹੱਸਣਾ (TT: hassṇā) (T: laughing).

- **Causative relation:** The causative relation shows interdependency between the two verbs. The causative verb relation is typically a phenomenon of Indian languages. The causative forms in the Punjabi WordNet are linked to their base root form. For example, the verb {ਖਵਾਉਣਾ} (TT: khavāuṇā) (T: to make someone to eat) is a causative verb of ਖਾਣਾ (TT: khāṇā), (T: eat).
- **Ability link:** It specifies the intrinsic features of a nominal concept. For example, the verb ਤਰਨਾ (TT: tairnā) (T: swim) is the intrinsic ability of the noun ਮੱਛੀ (TT: macchī) (T: fish).
- **Capability link:** The capability link specifies the acquired features of a nominal concept. For example, the verb ਤੈਰਨਾ (TT: tairnā), (T: swim) is the capability of the nouns ਆਦਮੀ (TT: ādmī), (T: person) or ਬੰਦਾ (TT: bandā), (T: person).
- **Function link:** The function link relation specifies the relation of a nominal concept. For example, the verbs ਪੜ੍ਹਾਉਣਾ (TT: paṛhāuṇā), (T: teach) and ਗਿਆਨ ਦੇਣਾ (TT: giān dēṇā), (T: teach) are the function of the nouns ਅਧਿਆਪਕ (TT: adhiāpak), (T: teacher) or ਗੁਰੂ (TT: guru), (TT: teacher) or ਮਾਸਟਰ (TT: māsṭar), (TT: teacher).
- **Attribute:** It shows the linkage between a noun and an adjective. For example, the adjective ਖੰਬਦਾਰ (TT: khambhdār), (TT: having wings) is an attribute of the nouns ਪੰਛੀ (TT: pañchī), (TT: bird) or ਪਰਿੰਦਾ (TT: parindā), (T: bird).
- **Modifies noun:** The adjectives that can only modify certain nouns are linked in the Punjabi WordNet by the relation modifies noun. For instance, the adjective ਸੁਪੁੱਤਰ (TT: suputtar) (T: son with a good character), can be used with nouns ਆਦਮੀ (TT: ādmī), (TT: person) and ਬੰਦਾ (TT: bandā), (T: person)
- **Modifies verb:** The adverbs that can only modify certain verbs are linked in the Punjabi WordNet by the relation modifies verb. For example, the adverb word ਕਦੇ (TT: kadē) (T: sometimes) can be used with ਕੰਮ ਕਰਨਾ (TT: kamm karnā), (T: to work).

To illustrate the relational structure of the Punjabi WordNet Figure 3.2 demonstrate the relations hyponym, meronym, hypernymy of the synset ਘਰ (TT: ghar) (Kaur, et al., 2010).

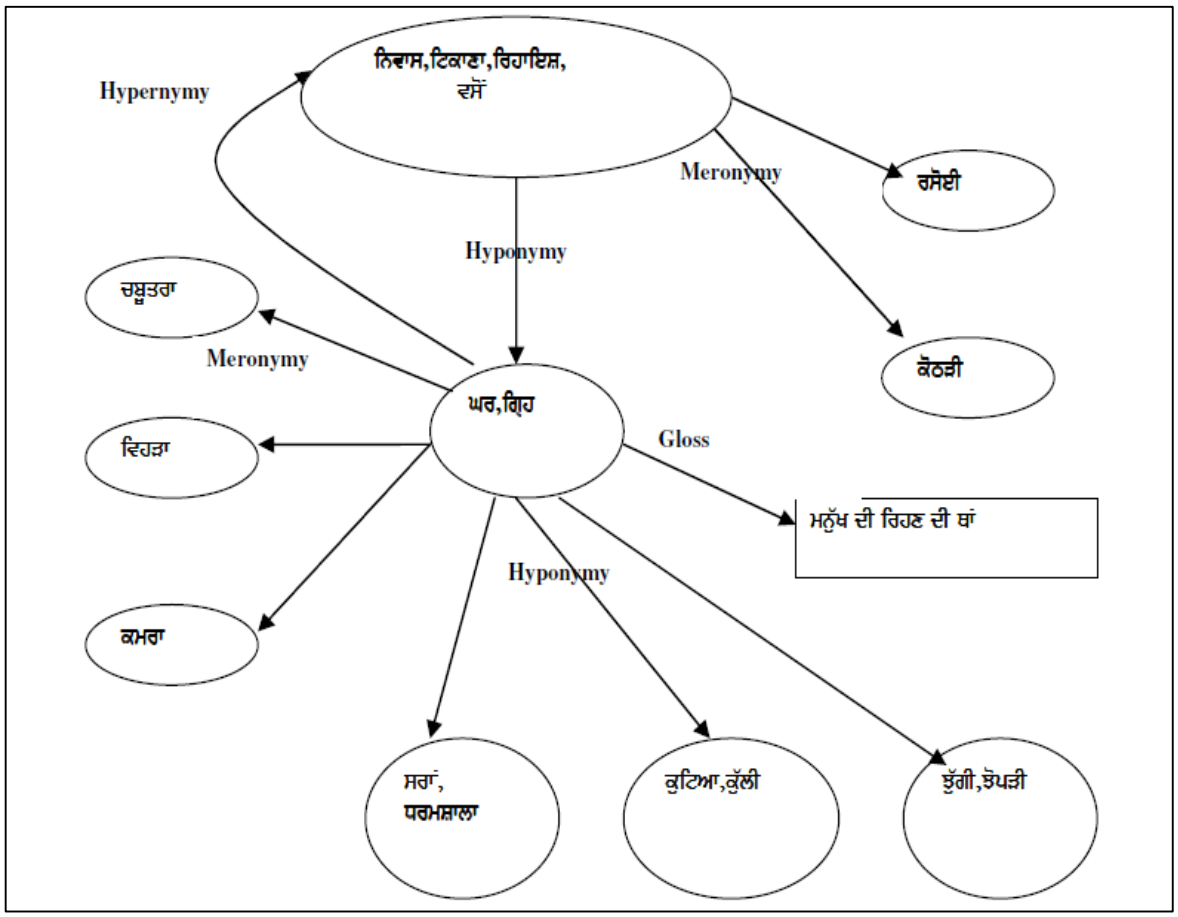


Figure 3.2: Relations of synset ਘਰ (TT: ghar) selected from the Punjabi WordNet (Kaur, et al., 2010)

The architectural details of the structured Punjabi WordNet show that it not only stores the synset that includes sense_id, PoS category, synonym words, gloss or concept definition, example sentence, but it also contains additional information such as different types of relations among the synset's.

3.2.2 Analysis of the Punjabi WordNet

The expansion approach followed to develop the Punjabi WordNet maps the Hindi WordNet synset's with the Punjabi language. It has been found that most of the synsets of the Hindi WordNet has an existence in the Punjabi language because both the languages belong to the Indo Aryan language family (Kaur, et al., 2010). In the expansion approach, the Hindi synset has been mapped to the Punjabi synset using a tool IL-MultiDic developed by IIT Bombay (Narang, et al., 2013).

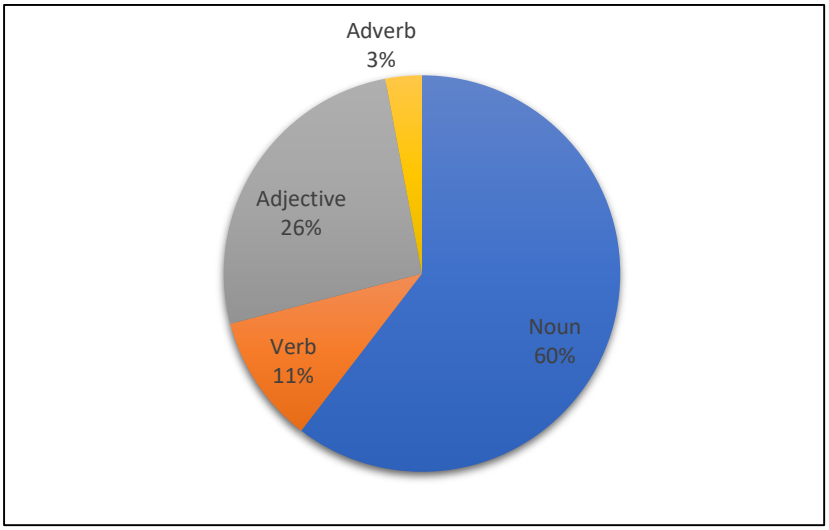


Figure 3.3: Distribution of PoS word classes in the Punjabi WordNet

The Punjabi WordNet consist of the monosemous (having single meaning) and polysemuous (having multiple meaning) Punjabi synset's words. It consists of a total (38485) words out of which (31846) are monosemous and (6639) are polysemuous Punjabi words. The Punjabi synset's of the Punjabi WordNet database have been primarily classified into PoS classes *viz.* nouns, verbs, adverbs and adjectives. The distribution of the Punjabi synset's according to their PoS categories in the Punjabi WordNet is depicted in Figure 3.3. It has been observed from Figure 3.3 that the most dominating PoS class of the Punjabi synset's is the noun class.

Table 3.3: The details of words classes having different senses in the Punjabi WordNet

Noun (23258) in WordNet																		
Senses	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Words	19432	2460	758	321	143	70	42	18	6	4	2	-	-	-	1	-	-	1
Verbs (4036) in WordNet																		
Senses	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Words	3153	469	181	79	52	31	13	14	8	8	10	5	5	2	1	3	1	1
Adjectives (10046) in WordNet																		
Senses	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Words	8227	1190	362	132	60	31	16	11	5	5	3	2	2	-	-	-	-	-
Adverbs (1145) in WordNet																		
Senses	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Words	1034	85	17	4	2	1	1	1	-	-	-	-	-	-	-	-	-	-

Table 3.3 displays the number of synsets and their corresponding senses stored in the Punjabi WordNet. There are (23258) nouns, (4036) verbs, (1145) adverbs and (10046) adjectives classes of the Punjabi synset's given in Table 3.3. Mostly Punjabi words in the Punjabi WordNet are monosemous words. It has been observed from Table 3.3 that with the increase in the ambiguity or senses of the Punjabi synsets's the count of synset's decreases.

Out of four PoS categories of Punjabi words in the WordNet, the most ambiguous Punjabi noun words have been selected. In this proposed WSD system for the Punjabi language, out of 3826 noun ambiguous words, 150 Punjabi noun words have been carefully chosen. Additionally, the sense definitions, example instances of these ambiguous words have also been collected from the Punjabi WordNet. The sense definitions of these Punjabi words are very close and overlapping. The average sense of these 150 ambiguous Punjabi noun words is 6.45. To extract the distinguishing senses of the 150 ambiguous Punjabi noun word, the process of merging the fine senses to coarse sense has applied, and its details have explained in the following subsection.

3.3 Merge Fine Sense to Coarse Sense

The selected ambiguous words from the enumerative lexicon Punjabi WordNet overalpping senses that is, the senses are closely related. For instance, the ambiguous word ਅੱਕ (TT: akk) has four senses defined in the Punjabi WordNet, as shown in Table 3.4. The meanings of the word ਅੱਕ (TT: akk) described in ਅੱਕ_sense1, ਅੱਕ_sense2 and ਅੱਕ_sense3 sense id's are related and overlapping. It is even difficult for the human being to clearly distinguish among the three different senses (ਅੱਕ_sense1, ਅੱਕ_sense2 and ਅੱਕ_sense3) of the ambiguous word ਅੱਕ (TT: akk).

Table 3.4: Word senses of ਅੱਕ (TT: akk) defined in the Punjabi WordNet

Word_Sense	**Gloss**	**Example**
ਅੱਕ_sense1	P: ਅੱਕ ਦੀ ਜਾਤੀ ਦਾ ਇਕ ਬਹੁਸਾਲੀ ਪੌਦਾ (TT: akk dī jātī dā ik bahusālī paudā)	P: ਅੱਕ ਦਾ ਦੁੱਧ ਅੱਖਾਂ ਦੇ ਲਈ ਹਾਨੀਕਾਰਕ ਹੁੰਦਾ ਹੈ (TT: akk dā duddh akkhāṃ dē laī hānīkārak hundā hai)
ਅੱਕ_sense2	P: ਇਕ ਪ੍ਰਕਾਰ ਦਾ ਅੱਕ ਦਾ ਰੁੱਖ (TT: ik prkār dā akk dā rukkh)	P: ਅੱਕ ਤੇ ਸਫ਼ੇਦ ਫੁੱਲ ਲੱਗਦੇ ਹਨ (TT: akk tē safēd phull laggdē han)
ਅੱਕ_sense3	P: ਇਕ ਬਹੁਵਰਸ਼ੀ ਵਨਸਪਤੀ (TT: ik bahuvarshī vansaptī)	P: ਅੱਕ ਦੀ ਪੱਤੀ ਜ਼ਹਿਰੀਲੀ ਹੁੰਦੀ ਹੈ (TT: akk dī pattī zahirīlī hundī hai)
ਅੱਕ_sense4	P: ਉਹ ਭੂਮੀ ਜਿਹੜੀ ਸਮਾਜਿਕ ਜਾਂ ਧਾਰਮਿਕ ਰੂਪ ਨਾਲ ਕਿਸੇ ਨੂੰ ਦਿੱਤੀ ਗਈ ਹੋਵੇ (TT: uh bhūmī jihṛī samājik jāṃ dhārmik rūp nāl kisē nūṃ dittī gaī hōvē)	P: ਇਸ ਤੇ ਧਾਰਮਿਕ ਪ੍ਰਬੰਧ ਕੀਤੇ ਜਾਂਦੇ ਹਨ (TT: is tē dhārmik prabandh kītē jāndē han)

Punjabi WordNet does not distinguish homonym (unrelated meanings) and polysemy words with the help of the hierarchical structures of senses of the words like in dictionaries. Nevertheless, WordNet lists the distinct senses of the words. The senses of the ambiguous words defined in the WordNet are overlapping. The narrow sense distinctions defined in WordNet makes them less suitable for the automatic WSD. This problem of fine senses of the ambiguous words in the Punjabi WordNet has been handled by grouping them into coarse senses.

The second step of sense tagged dataset preparation is merging the fine senses of the selected ambiguous words to yield more discerning coarse senses. (Ide & Wilks, 2007) have observed that humans and machines can discriminate the coarse level distinctions. The clear distinction among senses of the words is essential for a classification task such as WSD. (Palmer, et al., 2007) have observed that general senses produced by combining fine senses without the loss of meaning have yielded better results. The researchers have applied different resources to shape the appropriate senses of the words from the English WordNet like WordNet Domains WNDomains (Gella, et al., 2014), Standard Upper Ontology Working Group SUMO (Niles & Pease, 2001), SuperSenses (Ciaramita & Altun, 2006). The researchers have further applied different approaches to produce the coarse senses of the ambiguous words having fine senses selected from the WordNet for the WSD.

(Navigli, 2006) has reduced the granularity of the WordNet sense inventory by mapping to a manually crafted dictionary encoding sense hierarchies, using the Oxford Dictionary of English. (McCarthy, 2006) has focused on deriving different word-sense groups to overcome the fine grained distinctions of WordNet. (Snow, et al., 2007) have explored a supervised approach to prepare the coarse senses of the ambiguous words of the WordNet and have compared their approach with the earlier unsupervised clustering techniques used for sense merging. (Havasi, et al., 2010) have performed the coarse WSD using a blending approach that combines SemCor, WordNet, ConceptNet and Extended WordNet. (Bhingardive, et al., 2014) have applied a similarity-based sense merging approach that has calculated the WordNet gloss similarity using the word embedding features.

The golden standard for coarse sense generation of ambiguous words is the manual evaluation of senses (Palmer, et al., 2007). The overlapping senses of the words are merged to create coarse senses by carefully analysing the different WordNet relationships such as hyponymy and hypernymy. The set of rules have been framed to merge the fine senses of the ambiguous

words into coarse senses based on the semantic relations. The details of the sense merging rules are as follows.

- Rule 1: In this rule, the synset members (synonym words) selected from the Punjabi WordNet of the ambiguous word have been examined. It compares the synset members of each sense with all other synset members of senses of the ambiguous word. If all the synset members of different senses are the same, it merges the senses into a single coarse sense.
- Rule 2: This rule also compares the synset members having two or more senses of the ambiguous word. It combines the two senses of the ambiguous word if they have a minimum of two common synset members.
- Rule 3: The semantic relations hypernymy and hyponymy of the ambiguous words have been extracted and compared. If two different senses have the same hypernymy and hyponymy relations, then senses are collapsed to generate a single sense.
- Rule 4: This rule examines the example sentences of the ambiguous word given in the WordNet. The senses of the ambiguous word, which has no example sentences, have been dropped.

These rules have been individually applied to the 150 ambiguous noun words of the Punjabi language. Table 3.5 shows the process of reduction of 9 fine senses of an example target word ਝਾੜ (TT: jhāṛ) to 3 coarse senses by the application of above-stated sense reduction rules.

Table 3.5: Applying sense reduction rules on the fine senses of word ਝਾੜ (TT: jhāṛ)

Fine Sense	**Synonym and Gloss**	**Remarks**
S1 (T: Shrub)	Synonyms: ਝਾੜੀਆਂ (TT: jhāṛīāṃ), ਝਾੜ (TT: jhāṛ), ਝਾੜਾਂ (TT: jhāṛāṃ) Gloss: P: ਛੋਟੇਪੇੜ ਪੌਦਿਆਂਦਾ ਸਮੂਹ (TT: chōṭē pēṛ paudiāṃ dā samūh)	Unique base sense 1
S2 (T: Scold)	Synonyms: ਝਿੜਕ (TT: jhiṛak), ਘੂਰ (TT: ghūr), ਫਟਕਾਰ (TT: phiṭkār), ਡਾਂਟ (TT: ḍāṇṭ), ਝਾੜ (TT: jhāṛ), ਤਾੜਨਾ (TT: tāṛnā), ਖਰੀ-ਖੋਟੀ (TT: kharī-khōṭī) Gloss: P: ਝਿੜਕਣ ਜਾਂਘੂਰਨ ਦੀ ਕਿਰਿਆ ਜਾਂ ਭਾਵ (TT: jhiṛkaṇ jāṃ ghūran dī kiriā jāṃ bhāv)	Unique base sense 2
S3 (T: Yield)	Synonyms: ਫਸਲ (TT: phasal), ਉਪਜ (TT: upaj), ਪੈਦਾਵਾਰ (TT: paidāvār), ਝਾੜ (TT: jhāṛ) Gloss: P: ਖੇਤ ਵਿੱਚ ਉਪਿਜਆ ਹੋਇਆ ਅੰਨ ਆਦਿ ਅਜੇ ਪੌਦਿਆਂ ਵਿੱਚ ਹੀ ਲੱਗਿਆ ਹੋਵੇ (TT: khēt vic upjiā hōiā ann ādi jō ajē pōdiāṃ vic hī laggiā hōvē)	Unique base sense 3
S4	Synonyms: ਝਾੜੀਆਂ (TT: jhāṛīāṃ), ਝਾੜ (TT: jhāṛ) Gloss: P: ਉਹ ਛੋਟਾ ਪੌਦਾ ਜਿਸ ਦੀਆ ਟਾਹਣੀਆ ਜਮੀਨ ਦੇਬਹੁਤ ਕੋਲ ਨਿਕਲ ਕੇ ਚਾਰੇ ਪਾਸਿਆ ਵੱਲ ਫੈਲਦੀਆਂ ਹਨ	This sense is merge with sense 1 by following rule 1

Fine Sense	Synonym and Gloss	Remarks
	(TT: uh chōṭā paudā jis dīā ṭāhaṇīā jamīn dē bahut kōlōṃ nikal kē cārē pāsiā vall phaildīā han)	
S5	Synonyms: ਝਾੜੀ (TT: jhāṛī), ਝਾੜ (TT: jhāṛ) Gloss: P: ਸੰਘਣੇ ਅਤੇ ਕੰਡੇਦਾਰ ਪੌਦੇ ਜਾਂ ਝਾੜੀਆਂ ਦਾ ਸੰਘਣਾ ਸਮੂਹ (TT: saṅghaṇē atē kaṇḍēdār paudē jāṃ jhāṛīāṃ dā saṅghaṇā samūh)	This sense is merge with sense 1 by following Rule 2
S6	Synonyms: ਝਾੜੀਆਂ (TT: jhāṛīāṃ), ਝਾੜ (TT: jhāṛ) Gloss: P: ਸੰਘਣੀ ਅਤੇ ਕੰਡੇਦਾਰ ਝਾੜੀ ਜਾਂ ਪੌਦਾ (TT: saṅghaṇī atē kaṇḍēdār jhāṛī jāṃ paudā)	This sense is merge with sense 1 by following rule 1
S7	Synonyms: ਝਾੜ (TT: jhāṛ) Gloss: P: ਕੋਲ-ਕੋਲ ਉਗੇ ਝਾੜਾ ਦਾ ਸਮੂਹ (TT: kōl-kōl ugē jhāṛā dā samūh)	This sense is merge with sense 1 by following rule 3
S8	Synonyms: ਝਾੜ (TT: jhāṛ), ਵੱਡੀ ਝਾੜੀ (TT: vaḍḍī jhāṛī) Gloss: P: ਕੱਢੇਦਾਰ ਜਾਂ ਫਜੂਲ ਦੇ ਪੌਦਿਆਂ ਦਾ ਇਕ ਸਮੂਹ (TT: kaḍḍhēdār jāṃ phajūl dē paudiāṃ dā ik samūh)	This sense is merge with sense 1 by following rule 2

The sense reductions rules have helped in the generation of discerning coarse senses for the 150 ambiguous Punjabi noun words. This dataset that contains coarse senses of the Punjabi words has been used to perform the Punjabi WSD using supervised classifiers. The average coarse sense of the 150 ambiguous Punjabi noun words is 4.6. A comparative graph between the fine and coarse senses of the 150 ambiguous Punjabi noun words are shown in Figure 3.4.

Figure 3.4 shows the fine senses and their corresponding coarse senses for the 150 ambiguous words. The primary y-axis represents the number of coarse senses of the 150 ambiguous words. The ambiguous Punjabi words are represented on the x-axis.

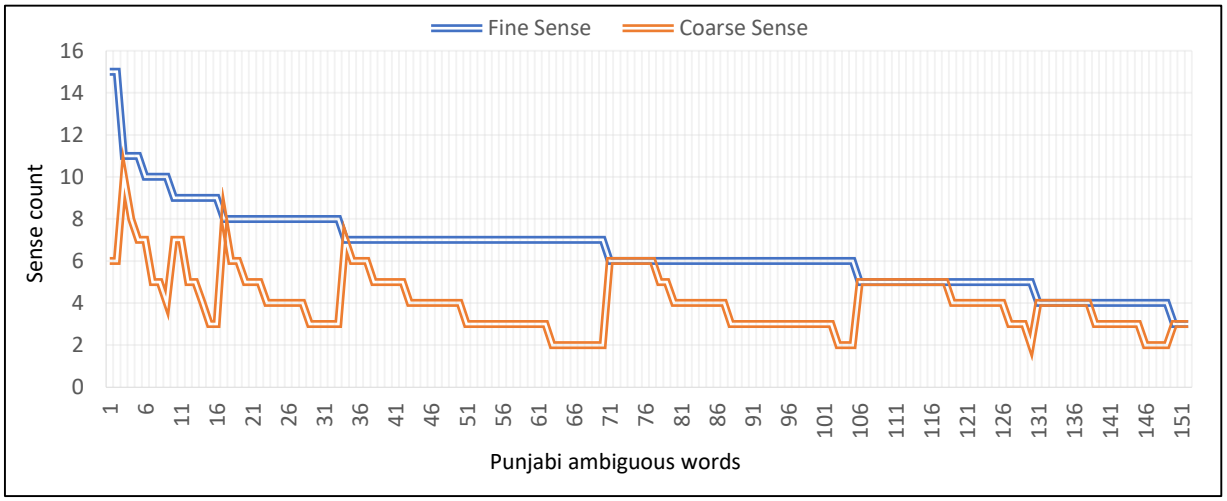


Figure 3.4: Fine and coarse senses of 150 ambiguous words

The distribution of words having different coarse senses across the 150 ambiguous words is shown in Table 3.6. The words having senses ranging from 2 to 8 exist in this dataset. It has

been observed from Table 3.6 that there are lesser number of Punjabi ambiguous noun words that have large number of senses.

Table 3.6:Distribution of 150 ambiguous words based on their senses

No. of Senses	Total Count	Percentage (%) of total dataset
2	26	17.3%
3	49	32.7%
4	34	22.7%
5	23	15.3%
6	14	9.0%
7	3	2.0%
8	1	0.6%

To prepare the sense tagged training dataset for the selected 150 ambiguous Punjabi noun words with coarse senses, the sense tagging process has been performed.

3.4 Sense Tagging Process

After selecting the ambiguous Punjabi words and defining the coarse senses for these words, the sense tagging process collects instances for each sense of the ambiguous words. The collected dataset has been validated using the inter-annotator agreement approach. The training datasets for the individual ambiguous word has been created. The details of data collection, inter-annotator agreement and Punjabi dataset are as follows.

3.4.1 Data Collection for the Ambiguous Words

The dataset for selected 150 ambiguous Punjabi words with coarse senses has been prepared by collecting the instances for different senses of words from the online and offline Punjabi language resources. This selection process and preparation of the lexical corpus of 150 Punjabi noun words have been performed manually. The objective of the sense-tagged corpora is to train/test the supervised WSD system based on the supervised machine learning classifiers. The performance of the supervised WSD system depends on the sense tagged corpus. Therefore, the sense-tagged corpora should ideally consist of all the senses of words (including rare senses), include various contexts, and provide information on the sense distribution in texts.

The sense tagged corpus for the Punjabi language was prepared using the targeted tagging approach. In the targeted tagging approach, the annotators tag all the occurrences of a single target ambiguous word with its different senses in the corpus. The targeted tagging approach prepares the sense tagged dataset quickly because the annotator has to focus only on a single target word and its senses at a time in the corpus (Domagoj & Snajder, 2016). The targeted tagging approach results in a lexical sample corpus. Lexical sample corpora cover various contexts and include almost all meanings of the target ambiguous word. The targeted tagging

approach has led to a balanced sense lexical corpus, which consists of all senses of the ambiguous words and different context.

The Punjabi lexical dataset is enriched by collecting the instances for the distinct senses of the selected ambiguous Punjabi words manually from online (newspaper websites, e-books, e-essay, blogs *etc*.) and offline (books, letters *etc*.) resources. The collected Punjabi text for the ambiguous words belongs to a wide variety of genres and domains. Each ambiguous word was searched on Google, Bing and Yahoo search engines. At least 100 different links have been explored to collect the instances for the different senses of the ambiguous word. The two independent annotators have segregated the gathered Punjabi instances of the ambiguous words into their different senses. The following instructions were given to the annotator.

- The two annotators should not consult each other during the annotation process.
- Annotators should distribute the instances only into the provided coarse senses of the ambiguous Punjabi word.
- Annotators can consult the Punjabi dictionary or Punjabi Wordnet to interpret the right sense of the ambiguous word in an instance.

The significant issues in the assignment of senses are the multiple sense assignment to the same instance and no sense assignment to instances. However, the disagreements have later resolved by an adjudicator for the final version of the corpus. The multi-sense assignment to an instance has been resolved by assigning the majority sense assigned by two of three annotators. The instances that have been assigned with no sense by two annotators are removed from the dataset. Table 3.7 shows the examples of the selection and rejection of instances for the ambiguous Punjabi noun words when a disagreement exists between the sense annotators.

The selection and rejection of the example instances for the sense tagged dataset of the five Punjabi ambiguous words are revealed in Table 3.7. For the first ambiguous word ਫਲ (TT: phal), the two annotators have assigned two different senses to the first instance. This multi-sense assignment issue has been resolved by selecting the majority sense of the ambiguous word. Annotators have assigned no sense for the second example sentence of the ambiguous word ਫਲ (TT: phal). Therefore, this example sentence has not been included in the sense tagged dataset of ਫਲ (TT: phal).

Table 3.7: Selection of sense instances based on annotator agreement

Word	Instance	Annotator1	Annotator2	Evaluator	Issue	Final
ਫਲ (TT: phal)	P: ਇਸ **ਫਲ** ਦੀ ਚਿੰਤਾ ਕਿਉਂ ਕਰ ਰਹੇ ਹੋ (TT: is phal dī cintā kiuṃ kar rahē hō)	Sense1 (outcome)	Sense2 (Fruit)	Sense1 (outcome)	Multi sense Assignment	**Sense1**
	P: **ਫਲ** ਨਾਲ ਹੱਥ ਕੱਟ ਗਿਆ (TT: phal nāl hatth kaṭṭ giā)	-	-	-	No sense assignment	**Reject**
ਹਾਰ (TT: hār)	P: ਰਾਮ ਨੂੰ **ਹਾਰ** ਪਾਇਆ ਗਿਆ (TT: rām nūṃ hār pāiā giā)	Sense1 (Garland)	Sense2 (ready)	Sense1 (Garland)	Multi sense Assignment	**Sense1**
	P: ਇਸਤਰੀ ਨੇ **ਹਾਰ** ਨੂੰ ਮੋਤੀਆਂ ਨਾਲ ਤਿਆਰ ਕੀਤਾ (TT: istarī nē hār nūṃ mōtīāṃ nāl tiār kītā)	-	Sense1 (Garland)	Sense1 (Garland)	Missing sense	**Sense1**
ਸੂਆ (TT: sūā)	P: **ਸੂਆ** ਲੱਗਣ ਕਰਕੇ ਸ਼ਾਮ ਦਾ ਤੇ ਪਾਸਾ ਹੀ ਰਿਹ ਗਿਆ (TT: sūā laggaṇ karkē shām dā tē pāsā hī rih giā)	Sense1 (knife)	Sense5 (injection)	Sense5 (injection)	Multi sense Assignment	**Sense5**
	P: ਅਮਨ ਨੂੰ ਰੋਟੀ ਖਵਾਉਂਦਿਆਂ ਮਾਂ ਦੇ ਹੱਥ ਤੇ **ਸੂਆ** ਵਜਿਆ (TT: aman nūṃ rōṭī khavāundiāṃ māṃ dē hatth tē sūā vajiā)	Sense 4 (tooth)	Sense5 (injection)	Sense1 (knife)	No agreement	**Reject**
ਝਾੜ (TT: jhāṛ)	P: ਸ਼ਾਮ ਦਾ **ਝਾੜ** ਨਾਲ ਹੱਥ ਸੁਜ ਗਿਆ (TT: shām dā jhāṛ nāl hatth suj giā)	Sense1 (shrub)	-	Sense1 (shrub)	Missing sense	**Sense1**
	P: ਇਸ ਵਾਰ **ਝਾੜ** ਦੀ ਗੱਲ ਨਾ ਕਰੋ ਜੀ ਬਹੁਤ ਘੱਟ ਹੋਇਆ ਹੈ (TT: is vār jhāṛ dī gall nā karō jī bahut ghaṭṭ hōiā hai)	Sense3 (yield)	Sense1 (shrub)	Sense3 (yield)	Multi sense Assignment	**Sense3**
ਜੱਗ (TT: jagg)	P: ਇਸ ਯੁਗ ਵਿੱਚ ਮੀਂਹ ਲਈ **ਜੱਗ** ਦੀ ਕੋਈ ਜਰੂਰਤ ਨਹੀਂ (TT: is yug vicc mīṃh laī jagg dī kōī jarūrat nahīṃ)	-	-	-	No sense assignment	**Reject**
	P: **ਜੱਗ** ਨਾਲ ਵਾਤਾਵਰਣ ਸਾਫ਼ ਹੋ ਜਾਂਦਾ ਹੈ (TT: jagg nāl vātāvraṇ sāpha◌ hō jāndā hai)	Sense1 (Ritual)	Sense2 (Jug)	Sense1 (Ritual)	Multi sense Assignment	**Sense1**

Similarly, for the second ambiguous word ਹਾਰ (TT: hār), the multi-sense assignment issue in the first example sentence has been resolved by selecting the majority sense. In contrast, for second example instance of ambiguous word ਹਾਰ (TT: hār), the missing sense issue has been updated by the evaluator.

In the third ambiguous word ਸੂਆ (TT: sūā) examples the issue multi-sense assignment in the first example sentence has been resolved by selecting the majority sense. In contrast, due to disagreement among the annotators, the second example sentence has been rejected. The issues of missing sense, multi-sense assignment and no sense assignment have handled similarly for the example sentences of the ambiguous words ਝਾੜ (TT: jhāṛ) and ਜੱਗ (TT: jagg).

During the annotation process, the average number of rejected Punjabi instances from the total dataset of 150 ambiguous words due to missing sense, multi-sense assignment and no sense assignment are 5%, 1.2%, and 6%. The collected sentences of ambiguous Punjabi words are stored in the plain text file. The independent text files are created for each sense of an ambiguous word. The name of a sense file for an ambiguous word commences with the name of the word, followed by its PoS category and the sense id. Each sense file contains the complete sentence, which includes the target word. The instances of ambiguous words can contain a single or multiple sentences accompanying the ambiguous target word. It has been ensured that the sense files should have a minimum of 5 instances for the rarely used senses of the ambiguous words. The instances of the sense files of independent ambiguous words have checked for redundancy. This check ensures the uniqueness of data in sense files and enables the classification process to overcome the overfitting. Table 3.8 shows the details of collected dataset of an ambiguous word ਜੱਗ (TT: jagg). It shows the file name, number of instances per sense file, percentage of sense instances in the dataset and an example instance selected from the individual sense file of the ambiguous word ਜੱਗ (TT: jagg).

Table 3.8: Details of the sense tagged files prepared for the word ਜੱਗ (TT: jagg)

File Name	**Number of instances**	**Percentage (%)**	**Example instance**
ਜੱਗ_n_Sense1.txt	90	47.6	P: ਮੀਂਹ ਨਾ ਪੈਣ ਕਰਕੇ ਪਿੰਡ ਦੀ ਪੰਚਾਇਤ ਨੇ ਜੱਗ ਕਰਨ ਦਾ ਫੈਸਲਾ ਕੀਤਾ (TT: mīṃh nā paiṇ karkē piṇḍ dī pañcāit nē jagg karan dā phaislā kītā)
ਜੱਗ_n_Sense2.txt	35	18.5	P: ਪਾਣੀ ਦਾ ਜੱਗ ਲਿਆਓ (TT: pāṇī dā jagg liāō)

File Name	Number of instances	Percentage (%)	Example instance
ਜੱਗ_n_Sense3.txt	40	21.2	P: ਸਿਆਣਿਆਂ ਕਿਹਾ ਹੈ ਕਿ ਇਹ ਜੱਗ ਮਿੱਠਾ ਅਗਲਾ ਕਿੰਨਾ ਡਿੱਠਾ (TT: siāṇiāṃ kihā hai ki ih jagg miṭṭhā aglā kinnā ḍiṭṭhā)
ਜੱਗ_n_Sense4.txt	24	12.7	P: ਇਰਾਕ 'ਚ ਫਸੇ ਭਾਰਤੀਆਂ ਦੇ ਮਾਮਲੇ 'ਚ ਚੰਡੀਗੜ੍ਹ ਵਿਖੇ ਦੀਵਾਨਗੀ ਦਿਖਾਉਣ ਵਾਲੇ ਸੰਗਰੂਰ ਤੋਂ ਲੋਕ ਸਭਾ ਮੈਂਬਰ ਭਗਵੰਤ ਮਾਨ ਨੇ ਦੋਸ਼ ਲਗਾਇਆ ਹੈ ਕਿ ਪੀ. ਟੀ. ਸੀ. ਨਿਊਜ਼ ਚੈਨਲ ਅਤੇ ਜੱਗ ਬਾਣੀ ਟੀ. ਵੀ. ਉਸ ਦੇ ਪਿੱਛੇ ਪੈ ਗਏ ਹਨ (TT: irāk 'c phasē bhārtīāṃ dē māmlē 'c caṇḍīgaṛh vikhē dīvāngī dikhāuṇ vālē saṅgrūr tōṃ lōk sabhā maimbar bhagvant mān nē dōsa◌ lagāiā hai ki pī. ṭī. sī. niūja◌ cainal atē jagg bāṇī ṭī. vī. us dē picchē pai gaē han)
Total	189	100	

3.4.2 Inter-Annotator Agreement of Sense Tagged Dataset

The Inter-annotator (or inter-coder) agreement has been evaluated to estimate the quality of the Punjabi language sense tagged corpus. A chance-corrected measure inter-annotator agreement Cohen's kappa (Ron & Poesio, 2008) have been applied for the Punjabi sense tagged dataset. Equation (3.1) calculates the Kappa score for the individual Punjabi noun word. The Kappa score in equation (3.1) is defined by the observed agreement (A_o) and the chance agreement (A_c). The observed agreement (A_o) is the percentage of annotations on which both annotators agree, whereas the chance agreement (A_c) is the percentage of chance of an agreement by annotators according to their class distribution.

$$Kappa = \frac{A_o - A_c}{1 - A_c} \quad \ldots (3.1)$$

$$A_c = \sum_{k \in K} P(c_A \mid k).(c_B \mid k) \quad \ldots (3.2)$$

The chance agreement is estimated in equation (3.2), where c_A and c_B represent the chance of an agreement between annotators *A* and *B* on the class *k.* It is described as the number of agreements of independent annotators *A* and *B* that would have been expected by chance for class *k.*

Table 3.9: Confusion matrix for the word ਫਲ (TT: phal) between two annotators

		Annotator 1				
Annotator 2	Senses	S_1	S_2	S_3	S_4	Total
	S_1	153	1	0	1	155
	S_2	4	27	0	0	31
	S_3	0	1	7	0	8
	S_4	3	0	0	3	6
	Total	160	29	7	4	200

A confusion matrix for the individual ambiguous word has been prepared to estimate the chance (A_c) and the observed (A_o) agreement between the authors. For instance, the Kappa statistic for the Punjabi word ਫਲ (TT: phal) having four senses and 200 sentences is estimated by evaluating the confusion matrix given in Table 3.9. Table 3.9 estimates the observed agreement A_o (0.95) between two annotators as the ratio of summation of diagonal instances (153+27+7+3=190) to the total number of the instance (200) of the word ਫਲ (TT: phal) in the confusion matrix.

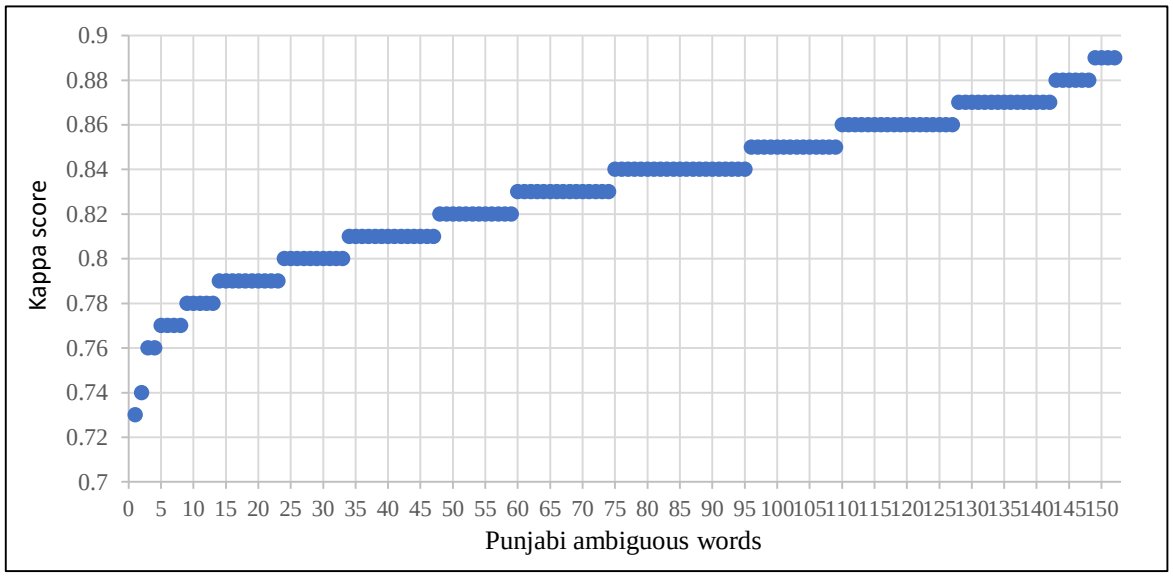


Figure 3.5: Kappa statistics of 150 ambiguous Punjabi noun words

Using equation (3.2), the chance agreement between the two annotators has been calculated for the word ਫਲ (TT: phal). The chance agreement (A_c) 0.64 is the ratio of summation of the product of senses determined by annotation 1 and 2 for four sense to the total number of instances of the sense tagged ambiguous word. It is obtained by substituting the confusion matrix values in equation (3.2) [(155x160+31x29+8x7+6x4)/(200*200)]. By substituting the observed (A_o) 0.95 and the chance (A_c) 0.64 agreement values in equation (3.1), the Kappa score of 0.87 has been obtained for the word ਫਲ (TT: phal) between the two annotators.

Similarly, the Kappa score has calculated individually for 150 ambiguous words. The average inter-annotator agreement for 150 ambiguous words is 0.846, which is a good agreement and is an indication of the reliability of annotations. Figure 3.5 shows the kappa estimated for 150 ambiguous Punjabi noun words.

3.4.3 Sense Tagged Punjabi Language Dataset

The details of the sense tagged dataset for 150 ambiguous target words are shown in Table 3.10. There is a total of 22153 training instances collected for the 150 ambiguous Punjabi noun words. Table 3.10 shows the selected ambiguous words, their senses, maximum, the minimum and total number of instances. The ambiguous Punjabi words in Table 3.10 are arranged according to their descending coarse senses.

Table 3.10: Details of sense tagged dataset collected for 150 ambiguous words

Word	Fine Sense	Coarse Sense	Max	Min	Total	Word	Fine Sense	Coarse Sense	Max	Min	Total
ਜੋੜ TT: jōṛ	11	8	92	8	363	ਕੁੰਡੀ TT:kuṇḍī	6	6	33	6	81
ਗੱਠ TT:gaṭṭh	9	7	127	6	242	ਘਾਟ TT: ghāṭ	7	5	60	31	225
ਜੋਰ TT: jōr	7	7	76	5	202	ਝਾੜ TT: jhāṛ	8	5	72	17	221
ਕਰਮ TT: karam	7	7	69	5	177	ਭਾਰ TT: bhār	5	5	106	13	217
ਭਾਗ TT: bhāg	15	6	93	17	328	ਬੈਠਕ TT:baiṭhak	8	5	113	5	211
ਪਰਦਾ TT:pardā	8	6	132	17	280	ਲਾਲ TT: lāl	5	5	110	7	209
ਮੂਲ TT: mūl	6	6	85	22	263	ਧੱਕਾ TT:dhakkā	5	5	77	5	200
ਭਾਅ TT: bhāa	6	6	117	5	205	ਉਲਟੀ TT: ulṭī	5	5	70	9	191
ਟਿੱਕਾ TT: ṭikkā	6	6	61	6	172	ਜੱਗ TT: jagg	7	5	90	24	189
ਤਾਰ TT: tār	6	6	89	5	161	ਵਾਰ TT: vār	7	5	80	7	187
ਘੋੜਾ TT:ghōṛā	6	6	82	6	149	ਵੰਡ TT: vaṇḍ	5	5	95	6	175
ਸੰਘ TT:saṅgh	7	6	44	4	133	ਕਰਾਰ TT: karār	5	5	84	5	153
ਘਰ TT: ghar	6	6	51	11	132	ਪੱਲਾ TT: pallā	5	5	62	10	152
ਸਿੱਲ TT: sill	6	6	34	5	112	ਧਿਆਨ TT:dhiān	5	5	45	5	144
ਜਾਲ TT: jāl	7	6	77	5	108	ਸੂਆ TT: sūā	5	5	55	8	142
ਤਲਾ TT: talā	9	6	48	5	100	ਕੁੜੀ TT: kuṛī	5	5	59	12	139
ਦਮ TT: dam	7	6	27	5	89	ਸਰ TT: sar	5	5	74	6	138

Word	Fine Sense	Coarse Sense	Max	Min	Total	Word	Fine Sense	Coarse Sense	Max	Min	Total
ਆਨੰਦ TT: ānand	5	5	45	6	133	ਤਾਲ TT: tāl	6	4	91	5	126
ਮੁੰਡਾ TT: muṇḍā	5	5	86	5	126	ਘਾਤ TT: ghāt	5	4	44	9	124
ਘੜੀ TT: ghaṛī	5	5	57	6	108	ਛੱਲ TT: chall	4	4	56	5	124
ਅੱਡਾ TT: aḍḍā	7	5	33	5	93	ਘੰਟੀ TT:ghaṇṭī	8	4	55	6	123
ਸ਼ਕਤੀ TT: shaktī	5	5	31	9	92	ਢਿੱਡ TT: ḍhiḍḍ	5	4	92	6	123
ਅੰਤ TT: ant	9	5	43	6	89	ਸਿੱਧੀ TT:siddhī	5	4	59	8	113
ਜੇਠ TT: jēṭh	5	5	34	9	73	ਦਾਈ TT: dāī	6	4	58	6	100
ਭਾਵ TT: bhāv	5	4	147	40	373	ਫੁੱਲ TT: phull	6	4	61	6	98
ਇਸਤਰੀ TT: istarī	4	4	209	6	264	ਝੂਮਰ TT:jhūmar	4	4	38	10	89
ਹਾਲ TT: hāl	7	4	100	11	207	ਜੂਨ TT: jūn	4	4	62	26	88
ਲਹਿਰ TT: lahir	7	4	83	7	202	ਜੀਭ TT: jībh	4	4	66	9	87
ਫਲ TT: phal	5	4	153	6	197	ਮਾਪ TT: māp	5	4	42	5	86
ਸੋਚ TT: sōc	5	4	75	11	190	ਟੀਕਾ TT: ṭīkā	5	4	55	5	83
ਘੜਾ TT: ghaṛā	9	4	101	20	188	ਸੂਤਰ TT: sūtar	5	4	52	7	83
ਗਿਣਤੀ TT: giṇtī	5	4	120	5	169	ਟੋਕਾ TT: ṭōkā	4	4	31	6	76
ਹਾਰ TT: hār	4	4	79	32	169	ਚੱਕਰੀ TT: cakkrī	4	4	29	12	71
ਸਭਾ TT: sabhā	5	4	67	24	161	ਖੁਮਾਰੀ TT:khumārī	4	4	26	6	54
ਭੋਗ TT: bhōg	7	4	52	21	160	ਖੂੰਟੀ TT: khūṇṭī	4	4	24	6	45
ਅਰਬੀ TT: arabī	4	4	77	20	160	ਵੱਟ TT: vaṭṭ	4	3	63	6	299
ਘੋਲ TT: ghōl	4	4	70	7	157	ਯੋਗ TT: yōg	6	3	129	40	226
ਸਥਿਤੀ TT: sathitī	4	4	71	16	142	ਚਾਹ TT: cāh	9	3	168	7	196
ਘਟਾ	7	4	88	8	136	ਠੇਕਾ	6	3	79	17	195

Word	Fine Sense	Coarse Sense	Max	Min	Total	Word	Fine Sense	Coarse Sense	Max	Min	Total
TT: ghaṭā						TT: ṭhēkā					
ਮੱਤ TT: matt	4	4	59	16	130	ਦੁੱਧ TT:duddh	4	3	140	25	195
ਸਾਹਿਤ TT: sāhit	3	3	154	11	195	ਸ਼ੰਕਾ TT:shaṅkā	6	3	74	8	90
ਜੰਗ TT: jaṅg	7	3	136	20	177	ਖਾਨਦਾਨ TT:khāndān	4	3	69	8	90
ਮੈਦਾਨ TT:maidān	3	3	65	14	156	ਮਾਵਾ TT: māvā	3	3	57	7	90
ਵਚਨ TT: vacan	5	3	85	20	149	ਬੋਟ TT: bōṭ	4	3	41	8	89
ਰੋਕ TT: rōk	7	3	57	37	140	ਸਹਿਮਤੀ TT:sahimtī	6	3	46	13	88
ਵਕਤ TT: vakat	8	3	114	11	137	ਸ਼ਹਿਰ TT: sahir	3	3	37	23	83
ਵੇਲ TT: vēl	3	3	68	13	137	ਆਗਿਆ TT: āgiā	6	3	42	5	75
ਛਾਪਾ TT: chāpā	3	3	109	8	134	ਬਾਬਾ TT: bābā	5	3	32	18	72
ਚਾਰਾ TT: cārā	5	3	75	10	133	ਵਿਚਾਰ TT: vicār	7	3	26	10	70
ਕੰਮ TT: kamm	6	3	75	10	131	ਬੋਲ TT: bōl	4	3	56	6	70
ਦਰ TT: dar	6	3	65	17	124	ਗਣੇਸ਼ TT:gaṇēsh	5	3	50	6	69
ਛੁਤ TT: chūt	3	3	64	6	120	ਵੇਲਾ TT: vēlā	6	3	31	10	67
ਅੱਕ TT: akk	7	3	56	8	113	ਫੈਸਲਾ TT:phaislā	6	3	50	6	63
ਉੱਲੂ TT: ullū	3	3	59	23	112	ਛਤਰੀ TT: chatrī	5	3	38	9	62
ਵਿਰੋਧੀ TT:virōdhī	7	3	66	17	111	ਜਾਂਚ TT: jāñc	7	3	25	14	54
ਸੰਗ TT: saṅg	3	3	68	11	111	ਉਪਦੇਸ਼ TT:updēsh	3	3	36	6	54
ਸ਼ੈਲੀ TT: saailī	5	2	52	13	108	ਕੁਮਾਰ TT:kumār	3	3	26	9	54
ਕੁੰਡ TT: kuṇḍ	4	3	57	8	103	ਚਮਚੀ TT: camcī	4	3	36	8	53
ਮੱਠੀ	3	3	46	22	101	ਜਟਾ	3	3	28	6	46

Word	Fine Sense	Coarse Sense	Max	Min	Total	Word	Fine Sense	Coarse Sense	Max	Min	Total
TT: maṭṭhī						TT: jaṭā					
ਬਾਰੀ TT: bārī	5	3	60	6	100	ਡੋਡਾ TT: ḍōḍā	5	3	22	6	45
ਰਸੋਈ TT: rasōī	3	3	86	6	99	ਜਾਗ TT: jāg	3	3	26	14	40
ਕਰ TT: kar	6	3	71	5	98	ਰਸ TT: ras	9	2	132	37	207
ਬਸਤੀ TT: bastī	4	3	51	5	93	ਦੌਰਾ TT: daurā	2	2	101	73	174
ਸਿੱਖਿਆ TT:sikkhiā	4	2	77	15	132	ਖੂਨ TT: kamī	3	2	40	39	79
ਜੰਗਲ TT: jaṅgal	5	2	110	14	124	ਲਗਾਮ TT: lagām	2	2	41	38	79
ਜਮਾਤ TT: jamāt	5	2	61	56	117	ਇਤਿਹਾਸ TT: itihās	2	2	55	21	76
ਹੱਲ TT: hall	2	2	107	9	116	ਆਰੰਭ TT:ārambh	6	2	69	6	75
ਉਬਾਲ TT: ubāl	2	2	70	42	112	ਗਊ TT: gaū	2	2	65	7	72
ਗ੍ਰਾਮ TT: grām	2	2	66	45	111	ਫਨ TT: phan	3	2	33	33	66
ਦਰਦ TT:darad	2	2	57	53	110	ਵਾਕ TT: vāk	2	2	43	21	64
ਦੁੱਖ TT:dukkh	7	2	68	39	107	ਰੱਜ TT: rajj	3	2	53	8	61
ਵਾਦੀ TT: vādī	2	2	91	14	105	ਭੜਥਾ TT:bharṭhā	2	2	37	24	61
ਦਿਹਾੜੀ TT: dihāṛī	7	2	89	10	99	ਸੰਵਾਦ TT: saṃvād	7	2	36	8	44
ਵਿਕਾਸ TT: vikās	7	2	53	36	89	ਝਾੜਾਂ TT:jhāṛāṃ	2	2	22	20	42
ਕਮੀ TT: kamī	6	2	73	14	87	ਟੋਹ TT: ṭōh	2	2	19	6	25

Out of 150 Punjabi ambiguous words there are 20 words that have 200 or more instances in their respective sense tagged dataset. There are 7 ambiguous word that have less than 50 instances in their respective sense tagged dataset. The word ਭਾਵ (TT: bhāv) with 3 coarse senses,

has maximum number of instances (373). The word ਟੋਹ (TT: ṭōh) with 3 coarse senses has minimum number of instances (25).

The maximum and the minimum coarse senses of the ambiguous Punjabi words are 8 and 2, as shown in Table 3.11. Where as the minimum number of instances for any coarse sense of the ambiguous words is 5. The number of average instances per sense of the target ambiguous words is 36.6, as shown in Table 3.11.

Table 3.11: Details of training dataset collected for 150 ambiguous words

Number of Words	**Number of Senses per word**			**Number Instances per sense**			**Total instance of dataset**
	Max.	**Min.**	**Avg.**	**Max.**	**Min.**	**Avg.**	
150	8	2	4.03	209	5	36.6	22153

The manually prepared sense tagged dataset of 150 Punjabi ambiguous words is essential for the development and implementation of the supervised Punjabi WSD system. This standard dataset has helped us in training and testing of the supervised machine learning classifiers.

Chapter Summary

The process of data collections for 150 ambiguous Punjabi nouns has been discussed in this chapter. Firstly, the ambiguous Punjabi words have been identified. Punjabi WordNet, a lexical knowledge base, has explored to find ambiguous Punjabi words. The most ambiguous 150 Punjabi noun words have been selected from the Punjabi WordNet along with their glosses, example sentences and relationship information. The detailed structure of the Punjabi WordNet and its features have explained in this chapter. The Punjabi WordNet defines very fine senses of ambiguous words. Secondly, the fine senses of the selected ambiguous Punjabi noun words have merged into the coarse senses using a sense merging algorithm. Finally, the dataset for the individual Punjabi ambiguous words has collected manually from the various online and offline resources of the Punjabi language. The two annotators have assigned the collected example sentences of the ambiguous words to the right senses, and an evaluator checks this sense tagging. The kappa statistic of 150 ambiguous noun words of the Punjabi language has been measured. A detailed analysis of the sense tagged dataset of the 150 ambiguous Punjabi noun words has been presented in this chapter. The objective of creating a sense tagged dataset of the Punjabi language is to provide a standard dataset that can be used to train, test and compare the Punjabi WSD task using the supervised classifiers.

Chapter 4 : The Architecture of Punjabi WSD

In this chapter, the architecture of the Punjabi WSD system has been discussed. The objective of the Punjabi WSD system is to determine the sense of the ambiguous Punjabi word independent of any NLP task. The proposed Punjabi WSD architecture is an explicit WSD system that considers only a single target word in the instance. The individual phases of the Punjabi WSD architecture, namely data collection, pre-process, parsing, feature extraction, and classification task, have been deliberated in this chapter.

Accordingly, the data collection that is the first phase of the Punjabi WSD architecture describes the process of creation of the sense tagged dataset for 150 ambiguous Punjabi noun words. The second phase of the architecture that is the pre-processing phase, removes the noise such as nonstandard fonts, punctuations and multiple spellings from the context of the sense tagged dataset of the Punjabi language. The parsing phase of this architecture has applied the Punjabi shallow parser on the sense tagged dataset to extract the root words (lemmas) and the part of speech of the Punjabi words. In the feature extraction phase, the selection of different features of the individual Punjabi ambiguous words from their respective sense tagged dataset was discussed. The conversion of the string features into real number vectors and the working of the supervised machine learning and the deep machine learning classifiers have been discussed in the classification phase of the architecture.

4.1 The Architecture of the Punjabi WSD

The WSD resolves the lexical ambiguity. It has helped in improving the performance of various Natural Language Processing (NLP) tasks. However, the WSD has been implemented as an implicit component in the NLP tasks. The implicit implementation of the WSD has limited the scope of the WSD for a single NLP task. In contrast, the explicit WSD has been explored by researchers, which is independent of any NLP task (Navigli, 2009). The explicit WSD can be integrated with other NLP tasks. The explicit or standalone WSD system directly addresses the different problems of the WSD, such as sense granularity problem, language-based pre-processing issues, sense tagged inventory preparation, and utilizing knowledge bases. Moreover, the explicit WSD system enables the researchers to measure and compare its performances. It also estimates the generalization capabilities for the fine senses of words.

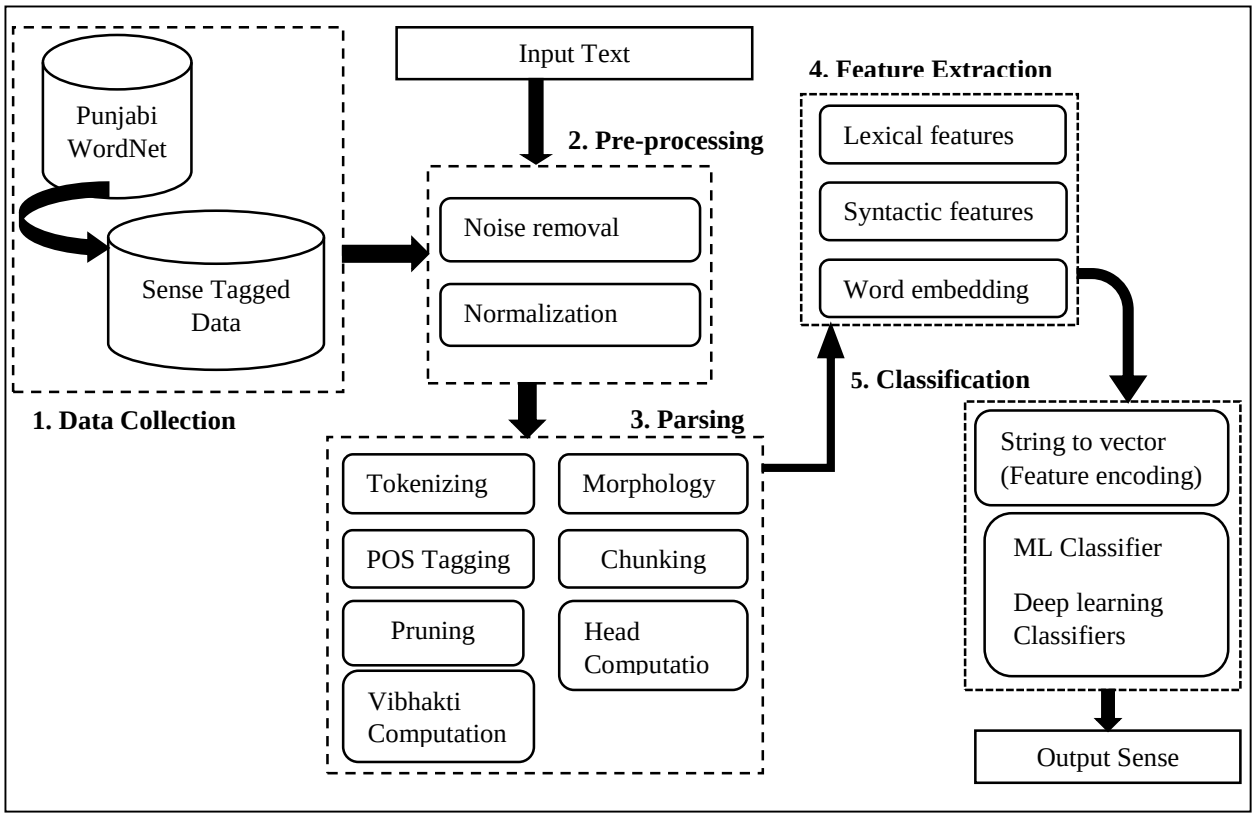


Figure 4.1:The Architecture of the Punjabi WSD

The proposed architecture of the Punjabi WSD is a standalone Punjabi WSD system. Its aim is to identify the sense of ambiguous Punjabi words from a given context. The supervised machine learning (ML) and deep learning classifiers are explored in this Punjabi WSD architecture to disambiguate the polysemous Punjabi words. This architecture follows the principle of the lexical sample WSD approach in which the WSD distinguishes the target words in the given context. The explicit Punjabi WSD architecture has been divided into five phases: data collection, pre-processing, parsing, feature extraction, and classification. Figure 4.1 describes the architecture of the explicit Punjabi WSD. The detailed functioning of the individual units of the Punjabi WSD architecture is as follows.

4.1.1 Data collection

The main aim of the data collection phase of the Punjabi WSD architecture is to prepare the sense tagged dataset for the supervised ML classifiers. The details of the collection and preparation of the sense tagged dataset of the most ambiguous 150 Punjabi words have mentioned in Chapter 3, titled Data Collection of the thesis. The prepared dataset acts as the input to the pre-processing unit of the Punjabi WSD architecture.

4.1.2 Pre-processing

The pre-processing phase performs the language-specific tasks such as neutralizing font variations, removing punctuations, digits, and normalization of the Punjabi text. The presence of unwanted text such as the English text, punctuations and digits in a sense tagged dataset of the Punjabi language is known as the noise. The pre-processing phase of the Punjabi WSD architecture has been applied individually on the dataset of 150 ambiguous noun words of the Punjabi language and has removed the noise from it. Removing noise such as the font variations, punctuation and the English text from the sense tagged dataset is as follows.

- **Font variation elimination:** During the sense dataset creation process, the Punjabi text has been collected from the different online and offline resources. Nevertheless, the fonts of the Punjabi text are not in standard Punjabi font (Unicode). The Punjabi text collected for sense tagged dataset is commonly written in Punjabi fonts like Anmol Liippi, Assess, Sutlej *etc*. The sense tagged dataset has been manually verified. If the Punjabi text is in a non-standard font, it has been converted into the Punjabi Unicode font with the help of the online font converter tools. In this work, the font converters by Punjabi University (http://fc.learnunjabi.org/font2Unicode.aspx) and by aglsoft web application (http://punjabi.aglsoft.com/punjabi/ converter/) have been used.
- **Punctuation removal:** After standardization of the Punjabi font of the context of the sense tagged dataset, the Punjabi context has been searched for the punctuation symbols and digits. The training dataset has been processed to eliminate the punctuations. The sentence-ending punctuation symbol (।) has been traced and removed from the Punjabi context.

  Similarly, the numeric digits and the English words present in the Punjabi sense tagged dataset has been removed. The individual characters of the dataset have been scanned to exclude punctuations, digits and English characters. This scanning procedure compares the characters of the dataset with the list of characters (such as punctuations, digits and English characters) and removes them.

Besides removing the unwanted text, punctuations and digits from the Punjabi text, the labelled text has normalized by replacing the abbreviations and spelling aberrations with the help of the Punjabi dictionaries.

- **Abbreviations completion:** The sense tagged dataset of the ambiguous Punjabi words consists of abbreviations or the short form. The abbreviated Punjabi tokens have been substituted by their complete forms using a manually prepared dictionary. For instance, the

abbreviations/acronyms ਡਾ., ਸ. and ਐਸ.ਜੀ.ਪੀ.ਸੀ. have been replaced with their respective full forms ਡਾਕਟਰ (TT: ḍākṭar), ਸਰਦਾਰ (TT: sardār) and ਸ਼੍ਰੋਮਣੀ ਗੁਰਦੁਆਰਾ ਪ੍ਰਬੰਧਕ ਕਮੇਟੀ (TT: shrōmṇī gurduārā prabandhak kamēṭī). The exapansion of Punjabi abbreviation ਸ. yields an ambiguous word ਸਰਦਾਰ (TT: sardār). The Punjabi abbreviations/acronym mapping dictionary has been prepared from the Punjabi corpus and web resources of the Punjabi language. A few abbreviations of the Punjabi language words and their complete forms are shown in Table 4.1. The mapping dictionary used in this work consists of 100 popular abbreviations/acronyms and their respective complete forms.

Table 4.1: Punjabi examples of abbreviations and their full forms

Acronym	**Full form**
ਡਾ	ਡਾਕਟਰ (TT: ḍākṭar)
ਸ	ਸਰਦਾਰ (TT: sardār)
ਐਸਜੀਪੀਸੀ	ਸ਼੍ਰੋਮਣੀ ਗੁਰਦੁਆਰਾ ਪ੍ਰਬੰਧਕ ਕਮੇਟੀ (TT: shrōmṇī gurduārā prabandhak kamēṭī)
ਯੂਪੀ	ਉਤਰ ਪ੍ਰਦੇਸ਼ (TT: utar pradēsh)
ਕਿਮੀ	ਕਿਲੋ ਮੀਟਰ (TT: kilō mīṭar)
ਭਾਜਪਾ	ਭਾਰਤੀ ਜਨਤਾ ਪਾਰਟੀ (TT: bhāratī jantā pāraṭī)
ਭਾਕਪਾ	ਭਾਰਤੀ ਕਮਿਊਨਿਸਟ ਪਾਰਟੀ TT: bhāratī kamiūnisaṭ pāraṭī
ਸੈਸਿ	ਸੈਕੰਡਰੀ ਸਿਖਿਆ (TT: saikṇḍarī sikhiā)
ਸਸਸਸ	ਸਰਕਾਰੀ ਸੀਨੀਅਰ ਸੈਕੰਡਰੀ ਸਕੂਲ (TT: sarkārī sīnīar saikṇḍarī sakūl)
ਪ੍ਰੋ	ਪ੍ਰੋਫੈਸਰ (TT: prōphaisar)

- **Spelling standardization:** There are multiple spellings for some Punjabi words in the Punjabi language sense tagged dataset. It is due to the usage of different online and offline resources of the Punjabi language to create the Punjabi sense tagged dataset. These resources have represented the Punjabi context using different Punjabi fonts, which have led to spelling deviations among the words. The conversions of the non-standard Punjabi fonts to the Punjabi Unicode font also lead to spelling aberrations. There are limited users of the standard Unicode keyboard of the Punjabi language. The non-availability of the standard auto-grammar checkers in word processing applications for the Punjabi words has also caused the multiple spelling problem of the Punjabi words in digital resources. The spelling correction pre-processing task is extremely important, to overcome the spelling

problems of the Punjabi language words. It ensures that only the unique Punjabi words should be selected as features.

Moreover, the selection of the wrong spelling of words causes the incorrect weight estimation of features and unnecessarily increases the computational requirements of the WSD system. The spelling deviations among the Punjabi words have been normalized by eradicating multiple spelling of the Punjabi words from the sense tagged dataset. The standard Punjabi spelling dictionary, which consists of 250 commonly occurring misspelt Punjabi words, has been prepared. The Punjabi spelling dictionary has been applied to the sense tagged dataset of the Punjabi language to perform the spelling corrections. Table 4.2 shows the list of a few Punjabi words having multiple spelling and standard spellings used in this work.

Table 4.2: Common misspelled Punjabi words

Misspelled Words	**Standard word**
ਊਹ	ਉਹ (TT: uh)
ਉਚੀ	ਉੱਚੀ (TT: uccī)
ਅਗੇ	ਅੱਗੇ (TT: aggē)
ਅਵਾਜ਼	ਆਵਾਜ਼ (TT: āvāz)
ਇਸੇ	ਏਸੇ (TT: ēsē)
ਸਚ	ਸੱਚ (TT: sacc)
ਸਾਫ	ਸਾਫ਼ (TT: sāf)
ਕੁਝ	ਕੁੱਝ (TT: kujjh)
ਤੇਜ	ਤੇਜ਼ (TT: tēz)
ਵਿਚ	ਵਿੱਚ (TT: vicc)
ਲਉ	ਲਓ (TT: laō)
ਫਿਲਮ	ਫ਼ਿਲਮ (TT: filam)

The pre-processing phase removes the noise (font variations, punctuations) and normalizes the Punjabi language context by completing abbreviations and standardizing the spellings of Punjabi words. The pre-processed text then passed to the parser phase to further refine the Punjabi language context.

4.1.3 Parsing

The Punjabi WSD architecture parsing phase extracts the root words (lemmas) from the sense tagged Punjabi dataset by removing postfixes and prefixes. The parsing phase also determines the Part of Speech (PoS) tags of the words present in a sense tagged Punjabi dataset. The

Punjabi shallow parser (developed by Consortium of Institutions headed by IIIT Hyderabad, India) has been used in this research work to generate the different base forms and PoS tags of the input instance words. There are seven intermediator stages of the Punjabi shallow parser: tokenization, morph analysis, PoS tagging, chunking, pruning, headword, and vibhikti computation. The output of these intermediator stages can also be used independently. The working of the Punjabi shallow parser and its different stages has been illustrated using an instance of the Punjabi language (Kumar & Sharma, 2012).

Punjabi example: ਜੰਗ ਦੇ ਸਮਾਜ ਉੱਤੇ ਬੁਹਤ ਬੁਰੇ ਪ੍ਰਭਾਵ ਪੈਂਦੇ ਹਨ।

Transliteration TT: jaṅg dē samāj uttē buhat burē prabhāv paindē han
English Translation: War has devastating effects on society.

(i) The first stage of the parser, i.e., tokenization, generates the atomic units of the input text. The atomic semantic unit of a sentence that is the sequence of characters is known as a token. The fundamental task of the tokenizer is to decompose the text or sequence of characters into basic units that cannot be further decomposed. The tokenizer converts input example sentence words, punctuation marks, and other symbols. The output of this intermediate step produces tokens with no information about the PoS of a word. The output of the tokenizer has produced the tokens of the Punjabi example but with the unknown PoS information (‘unk’).

Tokenizer:

<Sentence id="1">
1 ਜੰਗ unk
2 ਦੇ unk
3 ਸਮਾਜ unk
4 ਉੱਤੇ unk
5 ਬੁਹਤ unk
6 ਬੁਰੇ unk
7 ਪ੍ਰਭਾਵ unk
8 ਪੈਂਦੇ unk
9 ਹਨ unk
</Sentence>

(ii) The morphology stage determines the root words in the instance. The tokens from the previous stage are assigned with their grammatical features by the morphological analyser. The root word of the target word helps in narrowing down the search of its sense. The output of the morph analyser for the example sentence is as follows.

Morph:

<Sentence id="1">

1 ਜੰਗ unk <fs af='ਜੰਗ,n,m,sg,3,d,,'>|<fs af='ਜੰਗ,n,m,pl,3,d,,'>

2 ਦੇ unk <fs af='ਦੇ,v,any,sg,2,,0,0'>|<fs af='ਦਾ,psp,m,sg,,o,,'>|<fs af='ਦਾ,psp,m,pl,,o,,'>

3 ਸਮਾਜ unk <fs af='ਸਮਾਜ,n,m,sg,3,d,,'>|<fs af='ਸਮਾਜ,n,m,pl,3,d,,'>

4 ਉੱਤੇ unk <fs af='ਉੱਤੇ,adv,,,,d,,'>|<fs af='ਉੱਤੇ,psp,,,,d,,'>

5 ਬੁਹਤ unk <fs af='ਬੁਹਤ,unk,,,,,,'>

6 ਬੁਰੇ unk <fs af='ਬੁਰਾ,adj,m,sg,,o,,'>|<fs af='ਬੁਰਾ,adj,m,pl,,o,,'>

7 ਪ੍ਰਭਾਵ unk <fs af='ਪ੍ਰਭਾਵ,n,m,sg,3,d,,'>|<fs af='ਪ੍ਰਭਾਵ,n,m,pl,3,d,,'>

8 ਪੈਂਦੇ unk <fs af='ਪੈ,v,m,pl,any,,ਦਾ,ਦਾ'>|<fs af='ਪੈਂਦ,n,f,sg,3,o,ੀ◌1,ੀ◌1'>

9 ਹਨ unk <fs af='ਹੈ,v,any,pl,3,,,'>

</Sentence>

The feature structure ‘fs’ describe the grammatical features of each word. The composite attribute ‘af’ defines the seven attributes: root, lexical category, gender, number, person, case, tense, aspect and modality (TAM) information. The field is left blank if no value is given for a particular attribute. As shown in the above description, the word ‘ਜੰਗ’ (TT: jaṅg) has root word ‘ਜੰਗ’ (TT: jaṅg), lexical category noun (n), gender masculine (m), number as singular (sg), tense is third person, case direct and aspect and modality is not applicable to this word.

(iii) The third stage determines the PoS classes of the words, and it is called the PoS tagger. It marks each word of the sentence with its part of speech information like verb, noun, adverb, adjective. PoS tagger output for the example is as follows.

Pos tagger:

<Sentence id="1">

1 ਜੰਗ NN <fs af='ਜੰਗ,n,m,sg,3,d,,'>|<fs af='ਜੰਗ,n,m,pl,3,d,,'>

2 ਦੇ PSP <fs af='ਦੇ,v,any,sg,2,,0,0'>|<fs af='ਦਾ,psp,m,sg,,o,,'>|<fs af='ਦਾ,psp,m,pl,,o,,'>

3 ਸਮਾਜ NN <fs af='ਸਮਾਜ,n,m,sg,3,d,,'>|<fs af='ਸਮਾਜ,n,m,pl,3,d,,'>

4 ਉੱਤੇ NST <fs af='ਉੱਤੇ,adv,,,,d,,'>|<fs af='ਉੱਤੇ,psp,,,,d,,'>

5 ਬੁਹਤ JJ <fs af='ਬੁਹਤ,unk,,,,,,'>

6 ਬੁਰੇ JJ <fs af='ਬੁਰਾ,adj,m,sg,,o,,'>|<fs af='ਬੁਰਾ,adj,m,pl,,o,,'>

7 ਪ੍ਰਭਾਵ NN <fs af='ਪ੍ਰਭਾਵ,n,m,sg,3,d,,'>|<fs af='ਪ੍ਰਭਾਵ,n,m,pl,3,d,,'>

8 ਪੈਂਦੇ VM <fs af='ਪੈ,v,m,pl,any,,ਦਾ,ਦਾ'>|<fs af='ਪੈਂਦ,n,f,sg,3,o,ੀ◌1,ੀ◌1'>

9 ਹਨ VAUX <fs af='ਹੈ,v,any,pl,3,,,'>

</Sentence>

The words ‘ਜੰਗ’ (TT: jaṅg), ‘ਸਮਾਜ’ (TT: samāj) and ‘ਪ੍ਰਭਾਵ’ (TT: prabhāv) are classified as the nouns (NN) of the sentence. The word ‘ਪੈਂਦੇ’ (TT: paindē) and ‘ਹਨ’ (TT: han) are the

main verb (VM) and auxiliary verb (VAUX) respectively. One postpositions 'ਦੇ' (TT: dē) (PSP) is present in the example instance. There are two adjectives (JJ) in the example sentence. The noun location 'ਉੱਤੇ' (TT: uttē) (NST) is also tagged by the parser.

The PoS tags help in finding the right class of an ambiguous target word. The identification of the PoS class of an ambiguous word reduces the horizon of the search for its pertinent sense in an instance.

(iv) Chunker categorizes the group of words into noun, verb, adjective and adverb phrase. In this intermediate stage, the Punjabi shallow parser identifies noun phrases, verb groups, adjective phrase, and adverb phrase in a sentence. This syntactic information classifies the group of words into chunks and labels the chunks of the sentence. The output of the chunker, for example, the instance, is as follows.

Chunker:

```
--------
<Sentence id="1">
1       ((      NP
1.1     ਜੰਗ     NN      <fs af='ਜੰਗ,n,m,sg,3,d,,'>|<fs af='ਜੰਗ,n,m,pl,3,d,,'>
1.2     ਦੇ       PSP     <fs af='ਦੇ,v,any,sg,2,,0,0'>|<fs af='ਦਾ,psp,m,sg,,o,,'>|<fs af='ਦਾ,psp,m,pl,,o,,'>
        ))
2       ((      NP
2.1     ਸਮਾਜ    NN      <fs af='ਸਮਾਜ,n,m,sg,3,d,,'>|<fs af='ਸਮਾਜ,n,m,pl,3,d,,'>
2.2     ਉੱਤੇ    NST     <fs af='ਉੱਤੇ,adv,,,,d,,'>|<fs af='ਉੱਤੇ,psp,,,,d,,'>
        ))
3       ((      NP
3.1     ਬੁਹਤ    JJ      <fs af='ਬੁਹਤ,unk,,,,,,'>
3.2     ਬੁਰੇ    JJ      <fs af='ਬੁਰਾ,adj,m,sg,,o,,'>|<fs af='ਬੁਰਾ,adj,m,pl,,o,,'>
3.3     ਪ੍ਰਭਾਵ  NN      <fs af='ਪ੍ਰਭਾਵ,n,m,sg,3,d,,'>|<fs af='ਪ੍ਰਭਾਵ,n,m,pl,3,d,,'>
        ))
4       ((      VGF
4.1     ਪੈਂਦੇ    VM      <fs af='ਪੈ,v,m,pl,any,,ਦਾ,ਦਾ'>|<fs af='ਪੈਂਦ,n,f,sg,3,o,ੀ◌1,ੀ◌1'>
4.2     ਹਨ      VAUX    <fs af='ਹੈ,v,any,pl,3,,,'>
        ))
</Sentence>
```

Chunker classify the example sentence into four chunks that is ('ਜੰਗ ਦੇ' (TT: jaṅg dē), 'ਸਮਾਜ ਉੱਤੇ' (TT: samāj uttē), 'ਬੁਹਤ ਬੁਰੇ ਪ੍ਰਭਾਵ' (TT: buhat burē prabhāv)) three noun chunks 'NP' and one ('ਪੈਂਦੇ ਹਨ' (TT: paindē han)) verb chunks 'VGF'.

(v) The feature structure has been identified out of the different possible structures in the pruning stage of the Punjabi shallow parser. There are two types of sub-pruning, namely, morph pruning and pick one morph. Morph pruning picks feature structures

where the lexical category value matches the category value. Pick-one-morph selects only one feature structure based on the selection definition given to it. By default, it will pick the first feature structure. The output of the pruning stage of the Punjabi shallow parser for the Punjabi example using the pick one morph technique is as follows.

Pick One Morph:

<Sentence id="1">

1 ((NP

1.1 ਜੰਗ NN <fs af='ਜੰਗ,n,m,sg,3,d,,'>

1.2 ਦੇ PSP <fs af='ਦਾ,psp,m,sg,,o,,'>

))

2 ((NP

2.1 ਸਮਾਜ NN <fs af='ਸਮਾਜ,n,m,sg,3,d,,'>

2.2 ਉੱਤੇ NST <fs af='ਉੱਤੇ,adv,,,,d,,' poslcat="NM">

))

3 ((NP

3.1 ਬੁਹਤ JJ <fs af='ਬੁਹਤ,unk,,,,,,' poslcat="NM">

3.2 ਬੁਰੇ JJ <fs af='ਬੁਰਾ,adj,m,sg,,o,,'>

3.3 ਪ੍ਰਭਾਵ NN <fs af='ਪ੍ਰਭਾਵ,n,m,sg,3,d,,'>

))

4 ((VGF

4.1 ਪੈਂਦੇ VM <fs af='ਪੈ,v,m,pl,any,,ਦਾ,ਦਾ'>

4.2 ਹਨ VAUX <fs af='ਹੈ,v,any,pl,3,,,' poslcat="NM">

))

</Sentence>

(vi) Head word of the pruned chunks has been evaluated in this stage. The PoS tag of the computed head word to the chunk is assigned by the head computation step. A new feature head is assigned to chunk features as a head child named string. The head computation of the Punjabi example sentence is as follows.

Head Computation:

<Sentence id="1">

1 ((NP <fs af='ਜੰਗ,n,m,sg,3,d,,' head="ਜੰਗ">

1.1 ਜੰਗ NN <fs af='ਜੰਗ,n,m,sg,3,d,,' name="ਜੰਗ">

1.2 ਦੇ PSP <fs af='ਦਾ,psp,m,sg,,o,,'>

))

2 ((NP <fs af='ਉੱਤੇ,adv,,,,d,,' poslcat="NM" head="ਉੱਤੇ">

2.1 ਸਮਾਜ NN <fs af='ਸਮਾਜ,n,m,sg,3,d,,'>

2.2 ਉੱਤੇ NST <fs af='ਉੱਤੇ,adv,,,,d,,' poslcat="NM" name="ਉੱਤੇ">

))

3 ((NP <fs af='ਪ੍ਰਭਾਵ,n,m,sg,3,d,,' head="ਪ੍ਰਭਾਵ">

```
3.1     ਬੁਹਤ    JJ      <fs af='ਬੁਹਤ,unk,,,,,,' poslcat="NM">
3.2     ਬੁਰੇ    JJ      <fs af='ਬੁਰਾ,adj,m,sg,,o,,'>
3.3     ਪ੍ਰਭਾਵ  NN      <fs af='ਪ੍ਰਭਾਵ,n,m,sg,3,d,,' name="ਪ੍ਰਭਾਵ">
        ))
4       ((      VGF     <fs af='ਪੈ,v,m,pl,any,,ਦਾ,ਦਾ' head="ਪੈਂਦੇ">
4.1     ਪੈਂਦੇ   VM      <fs af='ਪੈ,v,m,pl,any,,ਦਾ,ਦਾ' name="ਪੈਂਦੇ">
4.2     ਹਨ      VAUX    <fs af='ਹੈ,v,any,pl,3,,,' poslcat="NM">
        ))
</Sentence>
```

(vii) The final output is generated by Vibhakti computation of shallow parser. This stage computes the final output of the shallow parser. The functions words of chunks have been added to the head words. Like function word 'ਵਿੱਚ' (TT: vicc) is combined with the head word 'ਜੰਗ' (TT: jaṅg). The ouput of the Vibhakti computation of the Punjabi example sentence is as follows.

Vibhakti Computation:

```
----------------------
<Sentence id="1">
1       ((      NP      <fs af='ਜੰਗ,n,m,sg,3,d,0_ਦਾ,' head="ਜੰਗ" vpos="vib1_2">
1.1     ਜੰਗ     NN      <fs af='ਜੰਗ,n,m,sg,3,d,,' name="ਜੰਗ">
        ))
2       ((      NP      <fs af='ਉੱਤੇ,adv,,,,d,0_ਉੱਤੇ,' head="ਉੱਤੇ" poslcat="NM" vpos="vib1_2">
2.1     ਸਮਾਜ    NN      <fs af='ਸਮਾਜ,n,m,sg,3,d,,'>
        ))
3       ((      NP      <fs af='ਪ੍ਰਭਾਵ,n,m,sg,3,d,,' head="ਪ੍ਰਭਾਵ">
3.1     ਬੁਹਤ    JJ      <fs af='ਬੁਹਤ,unk,,,,,,' poslcat="NM">
3.2     ਬੁਰੇ    JJ      <fs af='ਬੁਰਾ,adj,m,sg,,o,,'>
3.3     ਪ੍ਰਭਾਵ  NN      <fs af='ਪ੍ਰਭਾਵ,n,m,sg,3,d,,' name="ਪ੍ਰਭਾਵ">
        ))
4       ((      VGF     <fs af='ਪੈ,v,m,pl,3,,ਦਾ_ਹੈ+0,ਦਾ' head="ਪੈਂਦੇ" vpos="tam1_2">
4.1     ਪੈਂਦੇ   VM      <fs af='ਪੈ,v,m,pl,any,,ਦਾ,ਦਾ' name="ਪੈਂਦੇ">
        ))
</Sentence>
```

The parsed context of the sense tagged dataset has been passed to feature extraction phase of the Punjabi WSD architecture.

4.1.4 Feature Extraction

The feature extraction phase of the Punjabi WSD architecture selects the important clues from the sense tagged dataset of the Punjabi language. The context of the targeted ambiguous word has been used to obtain a different set of features. The context of a polysemous word is the sole mean to identify its meaning, and therefore, context is the pivot in determining the right sense

of the word. The context has been processed in the following ways to ascertain the meaning of the individual target word.

- Lexical features explore the context of ambiguous words in two ways, namely the local contextual features (collocation and n-grams) and the wider topical features (bag of words and co-occurrence). The local contextual features find the relational information between immediate neighbouring words and the target words. While the topical features collect the words surrounding the ambiguous words irrespective of the distance and their grammatical relations.
- Syntactic features discover the grammatical relations among the immediate neighbour words of the target ambiguous word in the context. The PoS tags of the sense tagged dataset have used as the syntactic features. The different combinations of the PoS tags have used as the syntactic features of the target ambiguous word.
- Word Embedding or distributional semantic features yield the fixed dimension feature for the target ambiguous words. It is a dense vector that is obtained by training a shallow neural network model. The d-dimensional vector of embedded words is the collection of real numbers. The closeness of vectors depicts the semantic relations between them. Besides the neural network predict based semantic features, the count-based semantic features have also prepared and explored using the co-occurring matrix.

The details of different features and their extraction algorithms have been described in chapter 5 of the thesis. The extracted features from the sense tagged dataset of 150 ambiguous noun words of the Punjabi language by feature extraction phase have passed to the classification phase of the Punjabi WSD architecture.

4.1.5 Classification

The classification phase of the Punjabi WSD architecture trains the machine learning (ML) and deep learning supervised classifiers individually on the selected features (lexical, syntactic and word embedding) of 150 ambiguous Punjabi language words. These trained classifiers determine the sense of the target ambiguous word in the test instance. Thus, the supervised classification task of the WSD architecture is a two-step process. In the first step, the supervised ML classifiers have trained on the individual feature set extracted using the sense tagged dataset of an ambiguous Punjabi noun word. This is known as the training of the classifiers. In the second step, the trained, supervised classifier predicts the sense of the ambiguous target word in the test instance. The second step is called testing. Training and testing of the ML

supervised classifiers have been achieved on the features selected from the sense tagged dataset of the ambiguous Punjabi words. However, the selected features are the collection of the Punjabi context words. The context words have to be codified in real values such that the ML classifiers can understand them. The string to vector process transforms string features into vectors with real numbers. The string to vector transformation and the training of the ML and deep learning classifiers are discussed in the following subsections.

4.1.5.1 String to Vector/ Feature Encoding

The String to vector (feature encoding) process transforms the selected text/word feature vector to a real values feature vector. The selected feature is compared with the text in the training dataset to create a vector file. This process of generation of vector file having real numbers from the text is known as the string to vector generation. The following are the two different string to vector transformations applied on the Punjabi string vectors.

- String to binary transformation converts the text training dataset of an ambiguous Punjabi noun word into a binary dataset. The unique feature sets extracted from the training data using the feature extraction methods are compared with the individual instances of its sense tagged dataset. The presence of the word in a word vector and in a training instance is indicated by 1 and 0 if it is absent. This comparison yields a binary matrix where the word vector act as column and instances of the sense tagged dataset of ambiguous word act as the rows. ML classifiers have used this matrix for their training.
- String to numeric transformations assign the frequency, term frequency (TF), term – inverse document frequency (TF-IDF) as weights to the words in the word vector according to their occurrence in the sense files of the ambiguous words. The selected weighted feature has been compared with sense tagged dataset to yield a matrix of real numbers. In the comparison process, if the word of feature vector does not exist in the instance it indicates 0. At the same time, the calculated weight is assigned to the word if it exits in the instance of the sense tagged dataset of the ambiguous word as well as in the feature vector.

The generated real number vector files of the ambiguous words have acted as the input to the ML classifiers for their training.

4.1.5.2 ML Classifiers Training

The process of automatic classification can be performed by the ML supervised classifiers using the different feature vectors. The supervised classifiers are trained on a training set S having n key-value pairs of features and the corresponding k classes. The feature space X is the

collection of real and discrete values called features *(x_1, x_2, …, x_m)* describing each example of *S.* The *k* classes or senses are the expected output attached to each example of *S*. The classifier learns a function *f* such that *k=f(x).* There are different learning functions like linear functions, decision trees, probabilistic functions, hyperplane functions *etc.* that have been applied for supervised learning approaches. Given a new vector, *x* the features of unlabelled test example, the trained classifier *f* predicts the *k* sense of the target word.

The supervised classifiers, namely Decision List (DL), Decision Tree (DT), Random Forest (RF), Naïve Bayes (NB), Support Vector Machine (SVM), k-Nearest Neighbour (kNN), AdaBoost (AB), Stacking (Stk) and StackingC (StkC) have been individually trained on different lexical, syntactic and word embedding features of 150 ambiguous noun words of the Punjabi language. Besides the ML classifiers, two deep learning classifiers, namely feedforward neural network or Multi Layer Perceptron (MLP) and Long Short Term Memory (LSTM) classifiers, have also trained on the word embedding features of the Punjabi language dataset. The functional description of ML classifiers applied for the Punjabi WSD is as follows.

(i) **Decision List**: The DL classifier is a sequence and conquer covering classifier that derives a list of weighted "if-then-else" ordered set of rules from the training dataset. The features of an ambiguous word (extracted from the sense annotated training dataset), sense of the word and a score constitute the rules in the DL classifier. DL classifier sorts the rules (feature, sense, score) in the decreasing order of their scores. Equation (4.1) calculates the score of a sense of an ambiguous word with respect to its features as the smoothed log of likelihood ratio of the probability of sense S_i given feature *f* divided by the sum of all likelihood probabilities of other senses given feature *f.* The score of sense S_i is estimated as the maximum among the feature scores.

$$score(S_i) = max_f \; log\left(\frac{P(S_i|f)}{\sum_{j\neq i} P(S_j|f)}\right) \quad \ldots (4.1)$$

The zero count and unreliable rules with negligible weight problems have been handled using smoothing and pruning. To disambiguate a word in a test example, the feature vector of the target word is extracted. The extracted feature vector is sequentially compared with the rules of the DL. The sense having the maximum estimated score calculated from the DL becomes the winner sense of the word in a test example. The hierarchical DL for sense disambiguation of the English language has yielded the best results in SENSEVAL-1 evaluation competitions (Yarowsky, 2000).

(ii) **Decision Tree**: The DT classifier follows a top-down approach to building a tree recursively. This divide and conquer method is worthwhile only when features have discrete values. Due to the feature-oriented structure of the DT, it is ideally suited for classification problems where a finite number of discrete features can easily and adequately describe the instances. The basic DT is shown in Figure 4.2, and its characteristics are as follows.

- Internal nodes of trees represent a feature.
- A branch represents the outcome of the test performed on internal nodes.
- Leaf nodes represent a sense of the word.

The DT's internal nodes and root node have been obtained by applying entropy and information gain concepts on the training dataset. To select the root node of the DT, information gain is calculated for each attribute (feature) of the training dataset. The attribute that shows the highest gain is selected as the root of the DT. The information gain is calculated using entropy. Entropy gives the idea of the distribution of instances as per the associated classes. Entropy is zero if instances belong to the same class and it is one if instances are distributed equally among the classes. The attribute with low entropy is needed so that it can distinguish between the classes clearly. In equation (4.2), the entropy calculation of the *K* attribute is given where *s* is the number of senses of the target word, p_i is the fraction of instances of *K* belonging to *i* sense.

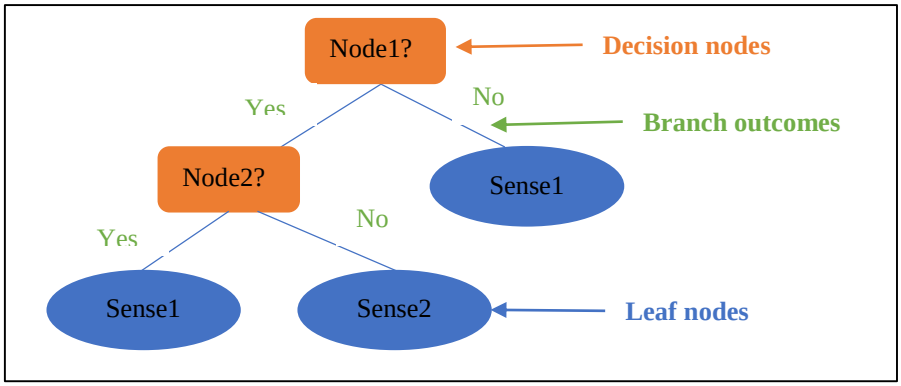


Figure 4.2: The basic decision tree structure

$$Entropy(K) = \sum_{i=1}^{s} -p_i * \log_2(p_i) \quad \ldots (4.2)$$

Information gain is defined as the difference in entropy of the parent node and the sum of entropies of all its child nodes. Equation 4.3 calculates the information gain (K, A) for an attribute A of a parent node associated with the set of examples K where *v* takes on the different values of A associated with the various branches emanating from the parent node. K_v stands for the subset of instances of *K*, which have value *v* for attribute A.

$$Gain(K, A) = Entropy(K) - \sum_{v \in Values(A)} \frac{K_v}{K} Entropy(K_v) \quad \ldots (4.3)$$

The attribute with the highest information gain is chosen to represent the node. This process recursively determines the next nodes and terminates when the nodes cannot be split further, or the information gain is zero. The Laplace smoothing functions have applied to handle the zero count problem. The Iterative Dichotomiser (ID3) is the basic DT algorithm as described above. The extended version of the ID3 algorithm is called C4.5, which includes the pruning reduced error technique to remove certain subtrees without affecting the accuracy of the classifier significantly (Witten, et al., 2011). To identify the sense of the target word in the test instance, its features are compared with the root of the DT and then subsequently with the following nodes till the leaf node or the sense of the word has reached. The DT classifier has been built and explored using the lexical and syntactic features of the English text (Mohammad & Pedersen, 2004a).

(iii) **Random Forest**: The RF classifier generates an ensemble of the multiple decision trees that work on bags of random samples. The majority classification determines the classification of the sense of the target word by the ensemble method. A single DT learning is sensitive to the training examples that is a slight variation in the training set may create significantly different trees. A few spurious instances might mislead the learning process creating an erroneous tree structure. To overcome these problems, multiple DT classifiers are prepared based on different sets of training data or samples. This technique is known as bagging, which generates multiple classifiers by random sampling of the training dataset with replacement. Quinlan has shown that the accuracy of the DT can be improved by bagging (Quinlan, 1996). The RF classifier based on bagging prepares the individual training dataset using replacement from the available training instances. The trees have been generated based on these different datasets. At each node of the tree, some subset of the features has randomly selected instead of examining all possible feature splits. By restricting each split-test to a small, random sample of features, the correlation between the trees has further minimized. Finally, the forest of DTs takes a vote. The pseudo-code of the RF classifier creation is as follows.

a) Select '*l*' features randomly from total '*p*' features of the dataset, such that '*l*'<< '*p*'.
b) Calculate the split point for '*l*' randomly selected features to produce root node.
c) Using best split create child nodes.
d) Repeat the first three steps until '*m*' nodes have produced.
e) Build the forest by repeating the four steps for '*n*' number of times to create '*n*' trees.

The RF classifier predicts the outcome of the test instance by using multiple DTs created. One vote is assigned to the classification of an instance by each DT classifier. The class which gets the maximum votes is chosen as the sense outcome for the test instance.

(iv) **Naïve Bayes (NB)**: The NB is a statistical classifier that estimates the probability distributions of classes in the training dataset. The NB classifier is based on the Bayes theorem, and it is the simplest probabilistic method that has been widely used for the WSD task (Singh & Kumar, 2018). In the context of the WSD, let there are *m* senses of an ambiguous word having $C = c_1, c_2, ..., c_n$ contextual features derived from its training dataset. The NB theorem in equation (4.4) calculates the posterior probability $P(s_i|C)$ of sense s_i in context *C*. According to equation (4.4), the *argmax* method maximizes the product of prior probability $P(s_i)$ of sense s_i with likelihood probability estimation $P(c_j|s_i)$ in evidence *C*. Similarly, posterior probability has estimated for all senses of the ambiguous word. The sense with the highest probability estimated by the NB classifiers is adjusted as the winner sense for the ambiguous word.

$$P(s_i|C) = argmax P(s_i) \prod_{j=1}^{n} P(c_j|s_i) \quad \ldots (4.4)$$

Equation (4.5) calculates the prior probability $P(s_i)$ of sense s_i of the ambiguous word. Prior probability $P(s_i)$ is the count of s_i sense (Ns_i) of the ambiguous word to the total count (N_{tw}) of its different senses the training dataset.

$$P(s_i) = \frac{Ns_i}{N_{tw}} \quad \ldots (4.5)$$

The likelihood probability $P(c_j|s_i)$ in equation (4.6) is estimated as ($N_{cj,si}$) the frequency of a context feature c_j occurring with the sense s_i in the evidence *C*.

$$P(c_j|s_i) = \frac{N_{cj,si}}{N_C} \quad \ldots (4.6)$$

The NB classifier assumes the independence of features among each other. Each contextual feature c_j independently contributes to probability estimation. The position of the features with respect to the target word does not matter. The likelihood probability is mostly zero because of the sparse presence of features in the training dataset. This anomaly is overridden using Laplace smoothing, *i.e.*, small quantity is added in numerator and denominator of likelihood probability.

(v) **Support Vector Machine (SVM)**: The SVM classifier learns the hyperplane that separates the example sets with maximum margin. It is a mixture of instance-based learning and linear modelling. The SVM represents binary features of training instances in

a plane. (Navigli, 2009). The SVM technique tries to find a hyperplane that separates the instances into discrete classes. Examples in the feature space that are nearest to the dividing hyperplane are known as the support vectors. Hyperplane that segregates maximum feature space instances into the positive and the negative set of examples is selected. Figure 4.3 shows the hyperplane equation $f(x) = w.x + b$ that geometrically separates two classes of examples.

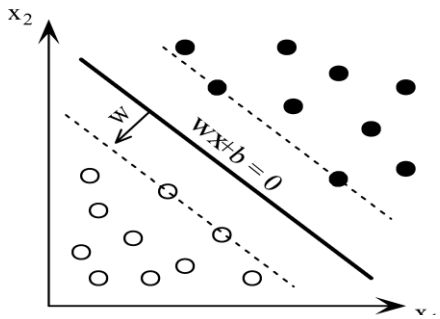


Figure 4.3: Hyperplane example for the SVM (Navigli, 2009)

The SVM is a binary classifier but for multiple classes classification such as WSD, it has to be adapted. The non-linear multiclass SVM classifier applies the kernel trick to transform the high dimensional distribution of the input points. The SVM classifier applies two approaches, one-against-all (OAA) and the one-against-one (OAO), to resolve the multiclass problem. In the OAA approach, the binary SVM distinguishes one class from all other classes and winner-takes-all strategy yield the final decision. In the OAO approach, the SVM is trained for all possible pairs of classes, and the final decision is obtained by majority voting. Training of the SVM algorithms is very computationally intensive and a slow process. The sequential minimal optimization (SMO) algorithm has been used to improve the performance of SVM (Witten, et al., 2011). The SVM classifiers have been the most successful classifier for English WSD (Zhong & Ng, 2010).

(vi) **k-Nearest Neighbour (k-NN)**: The exemplar-based or instance-based learning classifier is the simplest but very accurate classifier for the WSD. The k-NN is a supervised classifier that incrementally adds the feature vector of the sense tagged dataset into its feature space. The feature vector of the sense tagged dataset acts as the neighbours in the k-NN algorithm. The k-NN identifies a discrete sense of an ambiguous word by the majority vote of its neighbours. The test example set has compared with the set of learned vectors, and the sense of the most similar example vector has assigned to it as a winner sense. This learning-based technique uses memorized training dataset to classify the unknown instance.

Let $x = (x_1, ..., x_m)$ be the test instance vector in terms of its m features, and x_i are the stored example instances then the following steps are performed by the k-NN to identify the unknown instances

a) Set the k value a user define constant.

b) Calculate the distance metric between m attributes of x test instance and the stored m attributes of x_i instances using the Hamming distance. If the features of the training dataset and test instance are identical then difference between them is 0 else the difference is 1. The k-NN distance measurement has been calculated in equation (4.8). The weights w_j are assigned to j features and estimated employing gain ratio or modified value difference metric. Sort the estimated distances and determine the k minimum nearest neighbour.

$$\Delta(x, x_i) = \sum_{j=1}^{m} w_j \delta(x_j, x_{i_j}) \quad \ldots (4.8)$$

c) Collect the output class of the k nearest neighbours from the output class of the training vector.

d) Assign the prediction value to the query instance to majority of output class in which nearest neighbours belongs.

The right value of the k has been determined by performing cross-validation. The value of k is determined for an individual Punjabi ambiguous word. The k-NN algorithm has used different distance metrics like Hamming, Euclidian and modified value difference metric to measure the distance between neighbours (Navigli, 2009).

(vii) **Ensemble methods:** They combine the knowledge of the different ML classifiers to build a robust single classifier for the classification tasks. The ensemble methods try to eradicate the statistical, computational and representational problems of individual classifiers by combining them to attain better generalization (Rokach, 2010). The success of the ensemble system of the ML classifiers depends on the accuracy, independence of opinion, inherent diversity of base ML classifiers, and the combination methods. The ensemble methods are constructed in two phases: the base classifier generation phase and the combination phase. The objective of the base classifier generation phase of the ensemble methods is to generate accurate and diverse base classifiers. Two different approaches are followed to produce diverse classifiers: the training dataset manipulation and the classifiers type or parameter variations. In the combination phase of the ensemble methods construction, the outcomes of the diverse base ML classifier variants produced in the first phase have combined through two different methods, namely fixed rules and trained classifiers methods. The fixed rule combine strategy applies fixed vote or sum rules on the

outputs of base classifiers to produce the final output. While the trained methods strategy integrates the outputs of base classifiers using a meta classifier to produce the final output. The three ensemble methods applied on the Punjabi WSD task are as follows.

(viii) **AdaBoost:** This classifier is based on the training set resampling and rule-based combination approach of the ensemble method. The base classifier used in Adaboost is the DT. It is used to boost the performance of the DT. The AdaBoost works by putting more weight on difficult to classify instances and less on those already handled well. The basic AdaBoost algorithm deals with binary classification. However, to handle the multiclass WSD problem, the AdaBoost.M1 has been applied by the researchers.

(ix) **Stacking**: This ensemble classifier technique combines the output of heterogeneous individual machine learners using a combination arbitrator called the meta classifier.

(x) **StackingC:** The StackingC is an improved version of the stacking technique that uses the class probabilities as meta-data given by the base classifiers. The multi-response linear regression (MLR) most appropriate meta classier constructs a linear equation for each class and thus reduces the dimensionality of the attributes at the meta-level by a factor equal to the number of classes (Sesmero, et al., 2015). The improvement of StackingC over Stacking is due to the reduction of the meta-data's dimensionality and the high diversity of class models generated at the meta-level.

The architecture of the Punjabi WSD has not only implemented the ML classifiers but has also implemented the deep learning classifiers. The detailed functioning of the deep learning classifiers has been discussed in the following subsection.

4.1.5.3 Deep Learning Classifiers Training

Deep learning is a branch of the machine learning approach. Deep learning architectures have made a notable improvement in the fields of pattern recognition and computer vision in recent years (Young, et al., 2018). Consequently, the deep learning classifiers have increasingly focused on the research of the NLP. The ability of the deep learning classifiers to learn automatic feature representation and yielding superior results for the various NLP tasks have made them immensely popular among researchers. Deep learning classifiers are the neural networks (NN) that comprise a large and deeper number of processing layers. Early NNs have been defined with three layers; input, hidden, and output. To make the NN deep several numbers of hidden layers have added to it (Deng & Yu, 2014). This deep and complex NN enables the researchers to implement more complex, subtle and non-linear functions as they cannot be efficiently mapped with shallow NNs. The deepness of the NN is measured by the

number of hidden layers of the NNs. The successful implementation of the deep NN classifiers has attributed to the proliferation of cheaper graphical processing units, backpropagation algorithm and availability of the huge datasets.

Input to the deep NN is not a word, but it is the dense vector representation of a word called word embedding. The input has been prepared with the help of efficient algorithms that are capable of capturing very high-level features from the input data. Thus deep learning overcomes the need for handcrafted features by performing automatic feature learning. The deep NNs based on dense vector representations have produced superior results over the traditional machine learning for NLP tasks such as semantic role labelling (SRL), named-entity recognition (NER) and POS tagging (Collobert & Weston, 2008). Numerous complex deep learning NN algorithms have been employed to solve difficult NLP tasks (Young, et al., 2018). Broadly deep NN can be classified into feedforward neural network, Recurrent Neural Network (RNN), Radial Basis Function Neural Network, Kohonen Self Organizing Neural Network and Modular Neural Network (Shrestha & Mahmood, 2019). The RNN, MLP and LSTM deep learning models have successfully explored for the WSD task (Nithyanandan & C, 2019). The description of two different types of deep NN, namely the Multilayer feedforward NN and Long Short Term Memory (LSTM) investigated for the Punjabi WSD task, is as follows.

(i) **Multilayer feedforward network:** The multilayer feedforward NN is a supervised artificial neural network that consists of more than one hidden layer. In the feedforward networks, information runs in from the input to the output layer (via hidden nodes, if any) without any loops or circles. Figure 4.4 shows the basic structure of a feedforward NN or a multilayer perceptron with values and functions computed along the forward pass path. In a classification problem like WSD, the number of nodes in the input layer depends on the length of the input vector, while the number of output layer nodes depends on the pre-defined classes or senses. The hidden layer transforms the previous layer input with a weighted linear summation followed by the application of a non-linear activation function. The output layer transforms the last hidden layer into output values. The detailed description of the set of stacked layers of feedforward NN is shown in Figure 4.4 where *r* is the weighted sum of the inputs and *h* represents the non-linear activation function of *r* for each layer. The *W* represents the weights between the two units in the adjoining layers indicated by the subscript letters and *b* represents the bias value of the unit. The functioning of different layers of a feedforward NN is as follows

- The input layer comprises neurons that represent the unique set of words *(w_1, w_2, ..., w_n)* called vocabulary *V* extracted from the context of the sense tagged instances of the individual ambiguous target word. These words in vocabulary *V* are transformed into one hot binary representation. The one hot binary representation of a word is a binary vector of size *V* where index of the word is set 1 and all other values are zero.
- The hidden layer is the set of neurons that acts as a bridge between the input and output

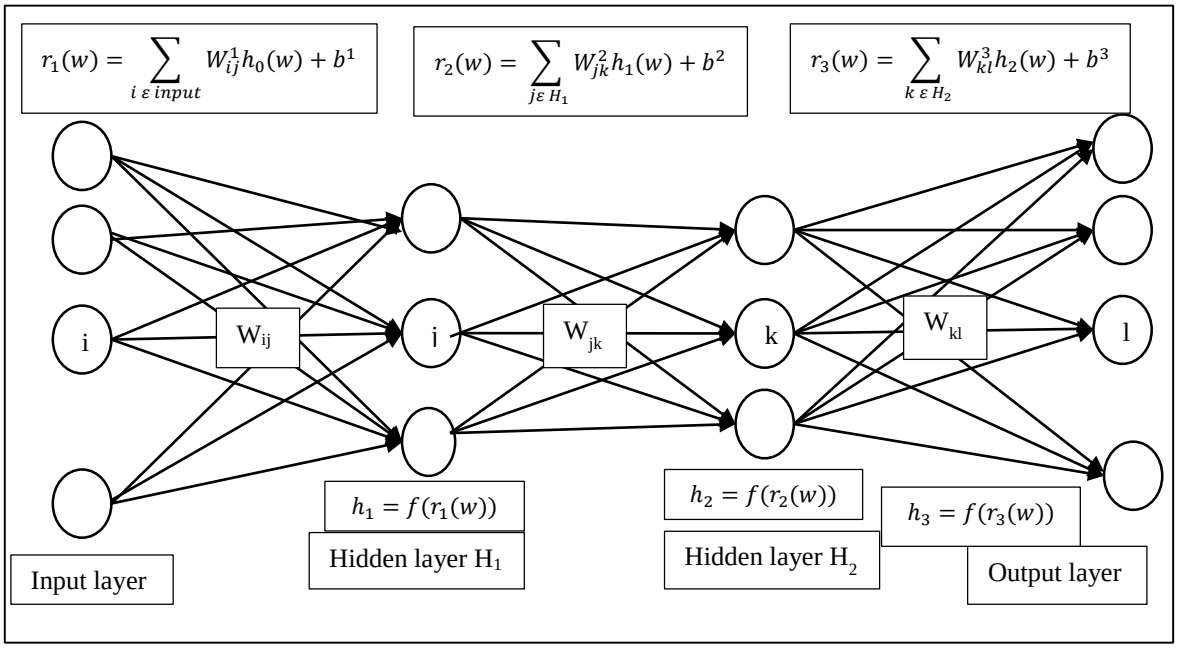


Figure 4.4: Multilayer feed forward NN

layer. The parameter weights W^1 of the hidden layer are applied to the input $h^0(w)$ (when k=1), which is one hot binary representation of words in the vocabulary. The pre-activation with k hidden layers of neural network where $k>0$ is estimated in equation (4.9).

$$r_k(w) = b^k + W^k h^{k-1}(w) \quad \ldots (4.9)$$

The hidden layer's neurons apply a nonlinear method to the summation of inputs $r_k(w)$ before passing it to the outgoing arrows. Commonly used activation functions by the hidden layers are *sigmoid*, hyperbolic tangent (*tanh*), hard *tanh* and Rectifier linear unit (*ReLU*) (Maas, et al., 2013). In this research work, the hidden layer activation applies the activation function ReLU as given in equation (4.10).

$$h_k(w) = f\big(r_k(w)\big) = max\big(0, r_k(w)\big) \quad \ldots (4.10)$$

- Output layer neurons yield the final classification results. The output layer applies the softmax method to derive the highest probability of the sense of the target word. For each batch of training examples softmax method as given in equation (4.11) is applied.

$$softmax(w_i) = \frac{e^{w_i}}{\sum_{j=1}^{n} e^{w_j}} \quad \ldots (4.11)$$

The weights of the network layers are regulated so that the error between the correct label and the predicted label can be minimized. The categorical cross-entropy function given in equation (4.12) has been applied to calculate the loss. This loss function L estimates the divergence between actual y^i and predicted $y'^{(i)}$ probability distributions.

$$L = -\frac{1}{n}\sum_{i=1}^{n}\left[y^i \, log \, y'^{(i)} + (1 - y^i) \, log\left(1 - y'^{(i)}\right)\right] \quad \ldots (4.12)$$

The error estimated from the forward flow information is reduced by the backpropagation that achieves the gradient of error for the given input (Rumelhart & McClelland, 1987). The final gradient (∇_{O_t}) calculation of a loss function is dependent on the gradient with respect to output layer ($\nabla_{r_P} L$) the gradient with respect to hidden layer ($\nabla_{h_k} L$) with activation function ($\nabla_{r_k} L$) such that(*0<k<P)* and the gradient with respect to weights and biases($\nabla_{W_k} L$, $\nabla_{b_k} L$). The backpropagation updates the weights of the parameters of the network as in equation (4.13).

$$O_{t+1} = O_t - \eta \nabla O_t \quad \ldots (4.13)$$

The gradient of the loss function is derived from the optimization algorithm Adam and η portion of it is reduced from the parameters of the network. The feedforward NN is a simple deep learning approach that explores the sense of the ambiguous word using all words of the dataset, ignoring the dependency relations among the context words.

(ii) **Long Short Term Memory**: The LSTM classifier is an extended Recurrent Neural Networks (RNN) classifier. Unlike feedforward classifiers where the NN layers are considered independent of each other, the processing units of the RNN classifier form a cycle. The RNN classifier can be viewed as a single layer network in which the output of the layer becomes the input to itself forming a feedback loop. As a result, the RNN classifier has memory about the previous states and which can be used to influence the current output. However, the training of the RNN classifiers has been considered complex due to the exploding/vanishing gradients problem (Shrestha & Mahmood, 2019). The backpropagation error gradients often grow too large or too small in long time series. LSTM

has been designed to selectively forget old states and focus on the current input state. The LSTM eliminates the inability of RNNs to capture long-term dependencies by adding a memory gate in its structure. The chain-like structure of the LSTM consists of a repeating module called memory cells. These memory cells tackle gradient diminishing and exploding problems. The memory cells of LSTM have linear dependence between the current c_t and its past c_{t-1} state. The state of the cell can be controlled by including or excluding past information. The basic gates of the LSTM are shown in Figure 4.5. The description of different internal units of the memory cell of the LSTM model is as follows.

- The forget gate controls the amount of previous time step information to be retained by a sigmoid layer that decides the information to be removed from the cell state. It assesses whether the past memory cell is useful for the computation of the current

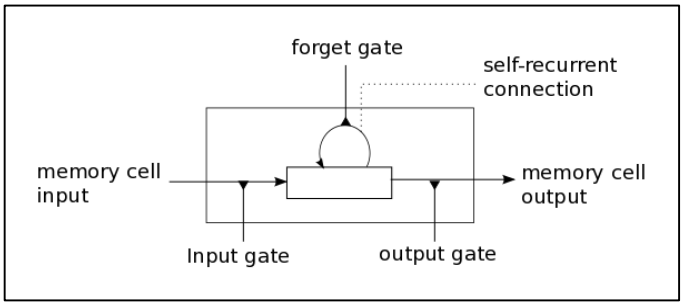


Figure 4.5: The basic gate of the LSTM

memory cell. It takes the sigmoid of the input w_t and the weights of previous timestamp h_{t-1} along with the bias b_f as shown in equation (4.14). The output of the sigmoid function determines to completely keep (if output =1) or completely discard (if output =0) the information.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, w_t] + b_f\right) \quad \ldots (4.14)$$

- The input gate determines the information be updated by applying the sigmoid function to the input word w_t, the past hidden state h_{t-1} and input bias b_i to determine whether or not the input is worth preserving. Equation (4.15) shows the input generated by the input gate. In the second step shown in equation (4.16), the *tanh* method is applied to the input to generate a new cell state or memory $\hat{c}$.

$$i_t = \sigma(W_i \cdot [h_{t-1}, w_t] + b_i) \quad \ldots (4.15)$$

$$\hat{c} = \tanh(W_c \cdot [h_{t-1}, w_t] + b_c) \quad \ldots (4.16)$$

$$c_t = f_t * c_{t-1} + i_t * \hat{c}_t \quad \ldots (4.17)$$

The final memory is estimated by the information to be neglected by forget gate output f_t and scaling of new information produced by input gate i_t and new cell state $\hat{c}$. The

summation of inputs from the forget and input gate shown in equation (4.17) are used to generate final memory.

- Output/Exposure gate decides which information of the cell state acts as the output. Its purpose is to separate the final memory from the hidden state. Equation (4.18) shows the sigmoid function produces the output o_t for the input word w_t with the last hidden state h_{t-1}. The hidden state of the cell is updated using output gate information o_t and the *tanh* of the estimated cell state.

$$o_t = \sigma(W_o \cdot [h_{t-1}, w_t] + b_o) \quad \ldots (4.18)$$

These deep learning classification models are trained on the sense tagged dataset, which is a set of instances with associated senses. The split validation approach has been applied to validate the MLP and LSTM classification models. The supervised machine learning classifiers have trained on the sense tagged dataset of the 150 ambiguos Punjabi noun words.

Finally, the output sense of classification module in the Punjabi WSD archirtecture predicts the sense of the Punjabi ambiguous noun word. The trained ML and deep learning models predict the sense of the ambiguous words present in the unseen test instances. The user can check the correctness of the predicted sense of the Punjabi ambiguous noun.

Chapter Summary

In this chapter, the explicit architecture of the Punjabi WSD is elaborated. It has been applied individually to the 150 ambiguous Punjabi noun words. The Punjabi language WSD is composed of five phases, namely data collection, pre-processing, parsing, feature extraction and classification. The data collection phase of this architecture describes the process of collection of the data for the 150 ambiguous Punjabi words. In the pre-processing phase, the prepared sense tagged dataset has been processed to remove the unwanted noise and the text from it. Parsing the third phase of the architecture further extracts the root words and PoS tags for the words in the sense tagged dataset. The feature extraction phase selects the lexical, syntactic and word embedding features from the parsed Punjabi language text. In the classification phase of the architecture, the conversion of string features to real number vectors and the training of the supervised ML and deep learning classifiers on the Punjabi language features have been discussed. The functional description of ML and deep learning classifiers have also been explained in this chapter. Finally, these trained models have been applied to predict the sense of ambiguous words in the Punjabi language test instances.

Chapter 5 : Features Selection

The features selection phase of the Punjabi WSD architecture has been discussed in this chapter. The context of the target ambiguous words has been processed to identify the useful evidence (the text features) from it. These text features have been divided into lexical, semantic and syntactic types. The categorization of the features is based on the different combinations of the context words and their respective grammatical relations. The different steps have been followed to extract the features from the context of the sense tagged dataset of the 150 Punjabi ambiguous words and from the unlabelled Punjabi corpus. The process of selecting the bag of words (BoWs) and co-occurring words has been described in the topical features section of this chapter. Similarly, the detailed process of selecting the local (collocations and n-grams), syntactic (part of speech tags) and semantic (count and predictive word embedding) features have been discussed in this chapter. The different steps of the generation of the predictive word embedding features, namely the collection of raw Punjabi corpus, pre-processing of the unlabelled text and the application of neural network models, have reported in this chapter. The Punjabi language word embedding features have also been tested against the similarity and analogy completion tasks presented in this chapater.

5.1 Feature Selection Steps

The feature selection process aims to extract the knowledge from the sense tagged dataset. The Machine Learning (ML) classifiers have trained using the extracted knowledge. The text features have been derived from the context of the ambiguous words. In the feature selection steps, the context of the target ambiguous word has been transformed into structured information, and the stop words have been eliminated from the context. The detail of the context representation of the Punjabi language context is as follows.

5.1.1 Context Representation

The context of the Punjabi language like a sentence, a paragraph or a document is the unstructured information for the ML classifiers. The input text has been processed to produce a structured format of text. The following steps have been performed to yield the structured text features.

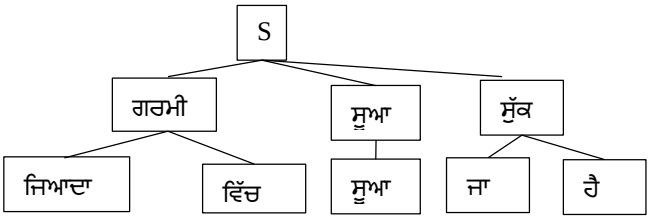
- Tokenization: A token is an instance of a sequence of characters in a sentence that is grouped as a useful semantic unit for processing. The fundamental task of the tokenizer is

to decompose the text or sequence of characters into basic units that cannot be further decomposed.

- Morphology: The morphological analyser identifies the root word and the other grammatical features of a word.
- Part of Speech (PoS) Tagger: It will mark each word of the sentence with its part of speech information like verb, noun, adverb and adjective.
- Chunking: Chunking is a process of dividing syntactically correlated structure. It involves identifying noun phrases, verb groups, adjective phrases and adverb phrases in a sentence. It also identifies the boundary of chunks and the label.
- Pruning: The pruning stage identifies the most appropriate feature structure out of different possible structures.

Table 5.1 shows the output of the individual step of the context processing methods using the Punjabi language illustration.

Table 5.1: Context processing of a Punjabi language sentence

ਜਿਆਦਾ ਗਰਮੀਆਂ ਵਿੱਚ ਸੂਆ ਸੁੱਕ ਜਾਂਦਾ ਹੈ। TT: jiādā garmīāṃ vicc sūā sukk jāndā hai T: In hot summer the pond dries	
Tokenization	(ਜਿਆਦਾ, ਗਰਮੀਆਂ, ਵਿੱਚ, ਸੂਆ, ਸੁੱਕ, ਜਾਂਦਾ, ਹੈ)
PoS Tagging	(ਜਿਆਦਾ\JJ ਗਰਮੀਆਂ\NN ਵਿੱਚ\PSP ਸੂਆ\NN ਸੁੱਕ\VM ਜਾਂਦਾ\VAUX ਹੈ\VAUX)
Lemmatization	(ਜਿਆਦਾ\JJ ਗਰਮੀ\NN ਵਿੱਚ\PSP ਸੂਆ\NN ਸੁੱਕ\VM ਜਾ\VAUX ਹੈ\VAUX)
Chunking	(ਜਿਆਦਾ\JJ ਗਰਮੀ\NN ਵਿੱਚ\PSP ਸੂਆ\NN ਸੁੱਕ\VM ਜਾ\VAUX ਹੈ\VAUX) NP NP VGF
Parsing	

The structured information selected from the context of the ambiguous words can be categorised into flat representations and tree/graph representations. The flat representations are due to the plain sense tagged training instances and are primarily explored by the supervised ML techniques. Whereas the tree or graph representations are generated from the interrelationships between the concepts extracted from the computational lexicons. The span of the tree or graph representations is the entire text of the computational lexicons. These representations are suitable for unsupervised and knowledge-based methods.

In this work, the flat representation of context has been explored. The context of the Punjabi sense tagged dataset has been normalized by eliminating the digits, the English language tags, punctuations, spelling anomalies, the font aberrations and abbreviations/acronyms words as explained in the previous chapter of this thesis. The pre-processing of the text helps in removing the redundancy, morphological inflections and deflections. The context of the Punjabi language has been further processed to remove the stop words from it.

5.1.2 Stop Word List

The stop word list is the collection of undiscriminating non-content words such as postpositions, articles, determinants, pronouns *etc.*, that do not influence the sense of the ambiguous target word. These postpositions, conjunctions, and pronouns have been removed from the sense tagged dataset of the 150 Punjabi ambiguous words before selecting the features. A manually prepared stop word dictionary of the Punjabi language has been used to remove such stop words. The stop word list has been prepared by analysing and selecting the postpositions, articles and pronouns from the sense tagged dataset. Table 5.2 shows a few examples of stop list words present in the Punjabi language stop word dictionary.

Table 5.2: Stop words of the Punjabi language

Category	**Examples**
Postpositions	ਨੇ(TT:nē), ਨੂੰ(TT:nūṃ), ਉੱਤੇ(TT:uttē), ਦਾ(TT:dā), ਤੋਂ(TT:tōṃ), ਵਿੱਚ(TT:vicc), ਉੱਪਰ (TT:uppar), ਨਾਲ(TT:nāl), ਰਾਹੀਂ (TT:rāhīṃ), ਦੁਆਰਾ (TT:duārā), ਵਾਸਤੇ (TT:duārā)
Pronouns	ਮੈਂ (TT: maiṃ), ਤੂੰ(TT: tūṃ), ਅਸੀਂ(TT: asīṃ), ਤੁਸੀਂ(TT: tusīṃ), ਉਹ(TT: uh), ਇਹ(TT: ih), ਉਹਨਾਂ(TT: uhnāṃ), ਇਹਨਾਂ(TT: ihnāṃ)
Conjunctions	ਅਤੇ (TT: atē), ਜਾਂ (TT: jāṃ), ਕਿ (TT: ki), ਜਦੋਂ (TT: jadōṃ),ਜਿਵੇਂ (TT: jivēṃ)

After the removal of the stop words from the normalized flat text representation of the context four different features namely topical, local, syntactic and semantic have been selected.

5.2 Types of Features

The different types of the features explored from the Punjabi language context are topical (BoWs, co-occurrence), local (n-grams, collocations), syntactic (part of speech tags) and semantic or distributional (count and prediction-based) features. Figure 5.1 depicts the different features selected from the Punjabi context. The topical, local and semantic features explore the different word combinations of the Punjabi context. Whereas the syntactic features collect the grammatical information from the context. The simple baseline count based most frequent sense feature has been discussed prior to topical, local, syntactic and semantic features.

5.2.1 Most Frequent Sense

The Most frequent sense (MFS) is the generalized count based feature of the ambiguous word. The MFS ranks the most commonly occurring sense in a sense tagged corpus independent of the context. The MFS of an ambiguous word can be calculated by counting the number of

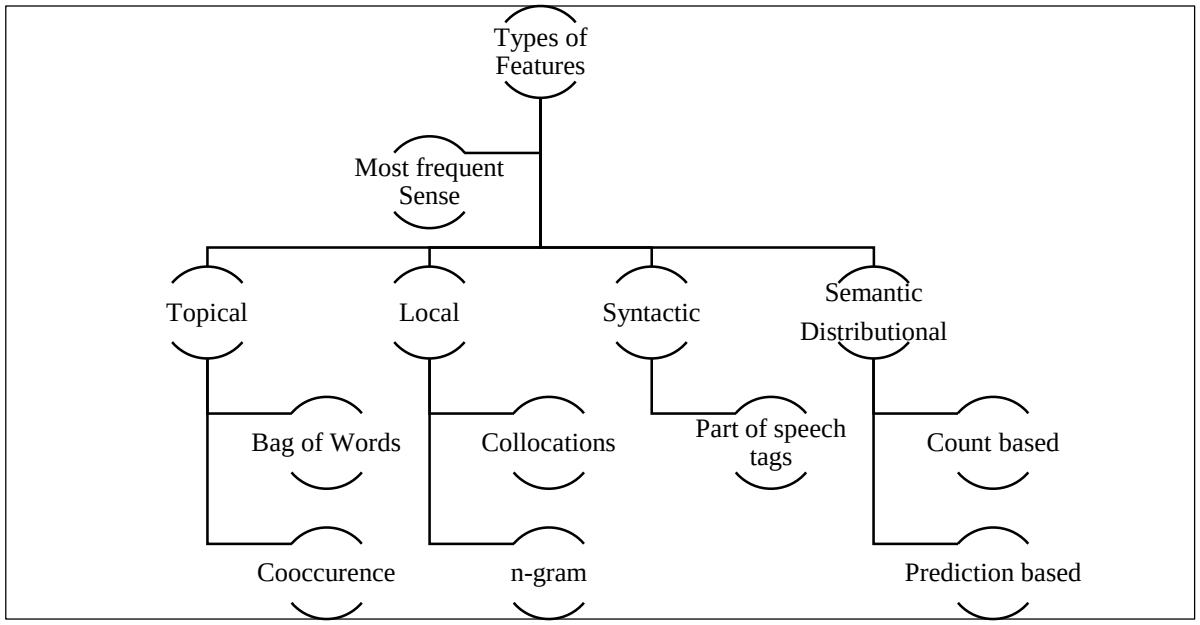


Figure 5.1: Different features selected for the Punjabi WSD

instances with the target word in various senses. This feature vector explores the skewed nature of senses of the ambiguous words. It has been observed that a few senses of ambiguous words tend to be used more commonly. While the other senses are very rare and only defined in the dictionaries. For instance, the word ਸੂਆ (TT: sūā) has five senses, as shown in Table 5.3. The sense 1 and sense 2 of word ਸੂਆ (TT: sūā) have frequently used in the Punjabi context. The sense 3 of word ਸੂਆ (TT: sūā) has been occasionally used. While the limited number of instances of the sense 4 and 5 of word ਸੂਆ (TT: sūā) indicate that these senses have limited usage in the Punjabi language context. Table 5.3 shows the frequency of the instances of the individual sense of the ambiguous Punjabi word ਸੂਆ (TT: sūā). The MFS acts as the baseline feature for the Punjabi WSD classification process. The performance of the different supervised ML classifiers using different features has been compared with the baseline MFS feature. Nevertheless, if all other features fail to confidently identify the intended sense, the most frequent sense has been assigned. Because the selection of the most frequent sense gives better accuracy than the accuracy obtained by a random choice (Preiss, et al., 2009).

Table 5.3: Number of instances corresponding to different senses of word ਸੂਆ (TT: sūā)

Different Senses	**Usage**	**Instances per Sense**
ਤਾਲਾਬ (TT: tālāb) (T: Pond)	Very frequent	55
ਚਾਕੂ (TT: cākū) (T: Knife)	Frequent	41
ਨਵਾਂ ਜੰਨ੍ਹਮਾ ਬੱਚਾ (TT: navāṃ jannhmā baccā) (T: New born of cow or buffalo)	Occasional	28
ਵੱਡੀ ਸੂਈ (TT: vaḍḍī sūī) (T: Large Injection)	Rare	10
ਤਿੱਖਾ ਸਖ਼ਤ ਮਜਬੂਤ ਦੰਦ (TT: tikkhā sakhaːt majbūt dand) (T: Sharp teeth, incisors)	Very rare	8

5.2.2 Topical Features

The topical features define the general topic of the text or discourse. The objective of these features is to capture the semantic domain of the text. The topic features are classified into BoWs and co-occurrence features. Topical features mainly include a sentence, a phrase, a paragraph or a document as a bag of words. The topical co-occurrence feature extracts the words that tend to occur frequently in the context of the target ambiguous word irrespective of the order and distance from the target word. The process of selection of BoWs and the co-occurring features from the ambiguous Punjabi words is as follows.

5.2.2.1 Bag of Words Features

The BoWs feature is a collection of words present in a sense tagged dataset of the ambiguous words. The order of occurrence of the words and their distance from the target word does not matter in the BoWs feature. Accordingly, each distinct word in the training dataset, constitutes an individual feature of the target ambiguous word. Table 5.4 illustrates the selection of BoWs features from the five different instances of the five independent senses of the Punjabi ambiguous word ਸੂਆ (TT: sūā). BoWs feature set ignores the exact ordering of words in the training dataset, and it considers only the unique words from the context, as shown in Table 5.4.

Table 5.4: Selected BoW features for different senses of word ਸੂਆ (TT: sūā)

Sense	Example instance	Bag of words per instance	Bag of words feature
S1	P: ਇਸ ਝਗੜੇ ਕਾਰਨ ਮੁਲਜ਼ਮਾਂ ਨੇ ਉਸ ਦੇ ਪਿਤਾ ਨੂੰ ਸੂਆ ਮਾਰ ਕੇ ਮਾਰ ਦਿੱਤਾ TT: is jhagṛē kāran mulzamāṃ nē us dē pitā nūṃ sūā mār kē mār dittā T: Due to this fight culprits have killed his father using a knife	ਇਸ, ਝਗੜੇ, ਕਾਰਨ, ਮੁਲਜ਼ਮਾਂ, ਪਿਤਾ, ਮਾਰ, ਸੂਆ, ਦਿੱਤਾ	ਇਸ, ਝਗੜੇ, ਕਾਰਨ, ਮੁਲਜ਼ਮਾਂ, ਪਿਤਾ, ਮਾਰ, ਦਿੱਤਾ, ਜਿਆਦਾ, ਗਰਮੀ, ਸੁੱਕ, ਰਿਹਾ, ਸਾਡੀ, ਗਊ, ਇਹ, ਪਹਿਲਾ, ਦੁੱਧ, ਦੰਦਾਂ, ਬਾਅਦ, ਬੱਚੇ, ਨਿਕਲ, ਆਇਆ , ਮੱਝ, ਲਗਾਉਣ, ਲਗਿਆਂ, ਡਾਕਟਰ, ਪਿਛਲੀਆਂ, ਲੱਤਾਂ, ਬੰਨ੍ਹ, ਦਿਤੀਆਂ
S2	P: ਜਿਆਦਾ ਗਰਮੀ ਦੇ ਕਾਰਨ ਇਹ ਸੂਆ ਸੁੱਕ ਰਿਹਾ ਹੈ TT: jiādā garmī dē kāran ih sūā sukk rihā hai T: Due to hot summer this pond dries.	ਜਿਆਦਾ, ਗਰਮੀ, ਕਾਰਨ, ਸੂਆ, ਸੁੱਕ, ਰਿਹਾ	
S3	P: ਸਾਡੀ ਗਊ ਦਾ ਇਹ ਪਹਿਲਾ ਸੂਆ ਹੈ TT: sāḍī gaū dā ih pahilā sūā hai T: This is the first baby of our cow	ਸਾਡੀ, ਗਊ, ਇਹ, ਪਹਿਲਾ, ਸੂਆ	
S4	P: ਦੁੱਧ ਦੇ ਦੰਦਾਂ ਤੋਂ ਬਾਅਦ ਬੱਚੇ ਦਾ ਸੂਆ ਨਿਕਲ ਆਇਆ ਹੈ TT: duddh dē dandāṃ tōṃ bāad baccē dā sūā nikal āiā hai T: After the milk teeth the child has a new incisor	ਦੁੱਧ, ਦੰਦਾਂ, ਬਾਅਦ, ਬੱਚੇ, ਸੂਆ, ਨਿਕਲ, ਆਇਆ	
S5	P: ਮੱਝ ਦੇ ਸੂਆ ਲਗਾਉਣ ਲਗਿਆਂ ਡਾਕਟਰ ਨੇ ਮੱਝ ਦੀਆਂ ਪਿਛਲੀਆਂ ਲੱਤਾਂ ਬੰਨ੍ਹ ਦਿਤੀਆਂ TT: majjh dē sūā lagāuṇ lagiāṃ ḍākṭar nē majjh dīāṃ pichlīāṃ lattāṃ bannh ditīāṃ T: Before giving injection to the buffalo the doctor has tied its back legs	ਮੱਝ, ਸੂਆ, ਲਗਾਉਣ, ਲਗਿਆਂ, ਡਾਕਟਰ, ਮੱਝ, ਪਿਛਲੀਆਂ, ਲੱਤਾਂ, ਬੰਨ੍ਹ, ਦਿਤੀਆਂ	

The BoWs string vectors consist of unique words selected from all the sense files of the ambiguous word. Therefore, the size of the BoWs feature is very large.

Algorithm 5.1 : Bag of words feature generation

```
/* Selecting a unique list of words from all the sense files of the ambiguous word w based on their occurrence count 2 */
input w, freq_min
output tokens_list
/*Prepare a list of all unique words and their count from the sense files of ambiguous words */
for each sense file of the w do
    separate each sentence of the w
    for each sentence do
        generate the word token by splitting sentences at " "
        if (token in stopword_list) then
          continue
        end if
        if (token exist in tokens_list) then
           tokens_list[token] += 1
        end if
    end for
end for
/*Remove the target ambiguous word*/
del token_list('w')
/*Select the words having frequency more than minimum threshold frequency*/
for each token in tokens_list do
    if count(tokens_list(token))<freq_min then
    tokens_list.remove(token)
    end if
end for
```

The selection process of the BoWs features has been described using Algorithm 5.1. In the first step, Algorithm 5.1 reads all the sense files of the ambiguous word and prepares the list of unique words along with their occurrence. After that, the stop words have been removed. The ambiguous target word has not been considered in the BoWs feature. Finally, Algorithm 5.1 selects the words having minimum threshold frequency (min=2) from the list of unique words prepared from the sense tagged dataset of the ambiguous word. Algorithm 5.1 has been individually applied to the 150 ambiguous Punjabi words to prepare the BoWs feature set.

5.2.2.2Co-occurrence Features

The co-occurrence features consist of suggestive words which occur in the context of the ambiguous word with a variable number of intermediary words between them. For example, the occurrence of word ਪਾਣੀ (TT: pāṇī) (T: 'water') in the context of an ambiguous word ਸੂਆ (TT: sūā) indicates its '*pond*' sense. Algorithm 5.2 defines the procedure to select the co-occurring words from the sense tagged dataset of the ambiguous Punjabi words. This algorithm scans the individual sense file of the ambiguous word and prepares the unique word count list. The words that occur more than the minimum value (determined for individual words) are selected as co-occurrence features. The minimum frequency parameter has been manually adjusted for the individual sense file of the ambiguous word. It is due to the skewed nature of the sense tagged dataset, and there are variable numbers of co-occurring words for an individual sense of the ambiguous word.

Algorithm 5.2: Co-occurring words generation

```
/* List of co-occurring words from all the sense files of the ambiguous word w */
Input freq_min
Output co_token_list
/*Read individual sense file of the ambiguous word*/
for each sense file of the w do
    for each sentence do
        generate the token by splitting sentences at " "
        for each token of sentence do
            count_word[token] += 1
        end for
    end for
/*Adjust manually the freq_min to remove the less occurring tokens */
    for each token in tokens_list do
        if count(count_word (token))<freq_min then
        count_word.remove(token)
    end if
co_token_list = count_word
count_word.clear()
end for
```

The large sense file has a higher value for the minimum frequency parameter, while the smaller sense file has a small value for the minimum frequency parameter. The values have been manually adjusted for the individual sense file of the ambiguous word. In Algorithm 5.2, the words having a frequency more than the minimum frequency has been selected. The co-occurrence features set of the ambiguous word has been prepared by combining the individual count list of the words collected from its individual sense file. Algorithm 5.2 has been individually applied to the 150 ambiguous Punjabi words to prepare their co-occurring feature set.

The BoWs and co-occurrence topical features induce domain relevance of the individual sense of the ambiguous words. However, to determine the dependence of the sense on the neighbourhood words, the local features have been selected and are discussed as follows.

5.2.3 Local Features

Local features are the words in the immediate vicinity of the target ambiguous word. The local features represent the fixed patterns of the local contextual information. These features include word forms and location with respect to the target word. There are two types of local features, namely collocation and n-gram features, that have been explored for the Punjabi WSD.

5.2.3.1 Collocation Features

These features are based on the assumption that neighbouring words provide strong clues about the target word (Mohammad & Pedersen, 2004a). The selected collocations are a combination of target words and neighbouring words. The collocation combinations of words are distinguished into compositional collocations or non-compositional collocations. In compositional collocations, the meaning of the target word can be derived from the meaning of the neighbouring words. While in non-compositional collocations, constituent words have a different meaning than the collocation itself, also called idiomatic compositions. In this work, we have selected the compositional collocations from the sense tagged dataset of the Punjabi language.

Collocations identify highly specific features that are the multiple word phrases. In collocation features, the number of words to the right or left of the target ambiguous word have determined by the parameter window size w_s. The window size w_s can be varied from 1 to 50. The small window size has been applied to select the collocation features (Lee and Ng, 2002). In this work, two window size, one, and two have been explored. The first collocation feature with window size one consists of one word left and one word right to the target word $w_{-1}; w_t ; w_{+1}$.

The second collocation feature with window size two (w_s =2) consist of two words left and two words right to the target word $w_{-2};w_{-1};w_t ;w_{+1};w_{+2}$. Table 5.5 demonstrates the selection of collocation features for the target ambiguous word ਸੂਆ(sūā) with window size 1 and 2.

Table 5.5: Collocation features of the word ਸੂਆ (TT: sūā)

Sense	**Example instance**	**Collocation with window size**	
		1	2
S1	P: ਇਸ ਝਗੜੇ ਕਾਰਨ ਮੁਲਜ਼ਮਾਂ ਨੇ ਉਸ ਦੇ ਪਿਤਾ ਨੂੰ ਸੂਆ ਮਾਰ ਕੇ ਮਾਰ ਦਿੱਤਾ TT: is jhagṛē kāran mulzamāṃ nē us dē pitā nūṃ sūā mār kē mār dittā T: Due to this fight culprits have killed his father using a knife	ਨੂੰ ਸੂਆ ਮਾਰ	ਪਿਤਾ ਨੂੰ ਸੂਆ ਮਾਰ ਕੇ
S2	P: ਜਿਆਦਾ ਗਰਮੀ ਦੇ ਕਾਰਨ ਇਹ ਸੂਆ ਸੁੱਕ ਰਿਹਾ ਹੈ TT: jiādā garmī dē kāran ih sūā sukk rihā hai T: Due to hot summer this pond dries.	ਇਹ ਸੂਆ ਸੁੱਕ	ਕਾਰਨ ਇਹ ਸੂਆ ਸੁੱਕ ਰਿਹਾ
S3	P: ਸਾਡੀ ਗਊ ਦਾ ਇਹ ਪਹਿਲਾ ਸੂਆ ਹੈ TT: sāḍī gaū dā ih pahilā sūā hai T: This is the first baby of our cow	ਪਹਿਲਾ ਸੂਆ ਹੈ	ਇਹ ਪਹਿਲਾ ਸੂਆ ਹੈ
S4	P: ਦੁੱਧ ਦੇ ਦੰਦਾਂ ਤੋਂ ਬਾਅਦ ਬੱਚੇ ਦਾ ਸੂਆ ਨਿਕਲ ਆਇਆ ਹੈ TT: duddh dē dandāṃ tōṃ bāad baccē dā sūā nikal āiā hai T: After the milk teeth the child has a new incisor	ਦਾ ਸੂਆ ਨਿਕਲ	ਬੱਚੇ ਦਾ ਸੂਆ ਨਿਕਲ ਆਇਆ
S5	P: ਮੱਝ ਦੇ ਸੂਆ ਲਗਾਉਣ ਲਗਿਆਂ ਡਾਕਟਰ ਨੇ ਮੱਝ ਦੇ ਪਿਛਲਿਆਂ ਲੱਤਾਂ ਬੰਨ੍ਹ ਦਿਤੀਆਂ TT: majjh dē sūā lagāuṇ lagiāṃ ḍākṭar nē majjh dīāṃ pichlīāṃ lattāṃ bannh ditīāṃ T: Before giving injection to the buffalo the doctor has tied its back legs	ਦੇ ਸੂਆ ਲਗਾਉਣ	ਮੱਝ ਦੇ ਸੂਆ ਲਗਾਉਣ ਲਗਿਆਂ

Algorithm 5.3 has been devised to determine two different sets of collocations from the sense tagged dataset of the individual Punjabi ambiguous word. Algorithm 5.3 traverses all sense files of the ambiguous word. It locates the ambiguous word in each instance of the sense file

Algorithm 5.3 Collocations selection

```
/* List of collocations selected from all the sense files of the ambiguous word w */
Input:  w, window_size
Output:  tokens_list
/*Traverse all sense files of the ambiguous word  w */
for each sense file of the w do
    separate each sentence of the w
    for each sentence do
        generate the word token by splitting sentences at " "
        Locate for the w
/* collect the collocation feature based on the window size */
        for each word in window_size around w do
            tokens_list.add(w-window_size , w, w+window_size)
        end for
    end for
end for
```

of the ambiguous word. The *window_size* parameter given in Algorithm 5.3 can be adjusted to determine the selection of the number of words to the right or left of the target ambiguous. In this research work, two values, 1 and 2 for *window_size*, has been applied. Finally, Algorithm

5.3 captures the collocation features in a list. The collocation features have been selected for the 150 ambiguous Punjabi noun words using Algorithm 5.3.

The selected collocation features have count based fixed phrases collocations, and they do not capture the flexible relations among the words. The flexible word relations have a variable number of words between them. To explore these flexible relations, n-gram features have been explored.

5.2.3.2 n-gram Features

The contiguous sequence of *n* words from a given text is known as n-gram features. The n-gram feature represents the patterns of the words occurring in the small window. The n-gram with *n* of the size one is called unigram, with *n* of the size two is called bigram, with *n* of the size three is called trigram and so on. The different n-gram features that have been explored in this research work are as follows.

- Unigram: The discrete words of the sense tagged dataset of the ambiguous words form the unigram features. Unigram features are the same as the BoWs feature set. Consequently, the unigram features consist of all words present in a sense tagged dataset of the ambiguous word irrespective of their order.

Table 5.6: Bigram features of word ਸੂਆ (TT: sūā)

Example instance	Bigram window size =2	Bigram window size =3	Bigram window size =4
P: ਜਿਆਦਾ ਗਰਮੀ ਦੇ ਕਾਰਨ ਇਹ ਸੂਆ ਸੁੱਕ ਰਿਹਾ ਹੈ TT: jiādā garmī dē kāran ih sūā sukk rihā hai T: Due to hot summer this pond dries.	ਜਿਆਦਾ ਗਰਮੀ, ਗਰਮੀ ਦੇ, ਦੇ ਕਾਰਨ, ਕਾਰਨ ਇਹ, ਇਹ ਸੂਆ, ਸੂਆ ਸੁੱਕ, ਸੁੱਕ ਰਿਹਾ, ਰਿਹਾ ਹੈ	ਜਿਆਦਾ ਗਰਮੀ, ਜਿਆਦਾ ਦੇ, ਗਰਮੀ ਦੇ, ਗਰਮੀ ਕਾਰਨ, ਦੇ ਕਾਰਨ, ਦੇ ਇਹ, ਕਾਰਨ ਇਹ, ਕਾਰਨ ਸੂਆ, ਇਹ ਸੂਆ, ਇਹ ਸੁੱਕ, ਸੂਆ ਸੁੱਕ, ਸੂਆ ਰਿਹਾ, ਸੁੱਕ ਰਿਹਾ, ਸੁੱਕ ਹੈ, ਰਿਹਾ ਹੈ	ਜਿਆਦਾ ਗਰਮੀ, ਜਿਆਦਾ ਦੇ, ਜਿਆਦਾ ਕਾਰਨ ਗਰਮੀ ਦੇ, ਗਰਮੀ ਕਾਰਨ, ਗਰਮੀ ਇਹ ਦੇ ਕਾਰਨ, ਦੇ ਇਹ, ਦੇ ਸੂਆ ਕਾਰਨ ਇਹ, ਕਾਰਨ ਸੂਆ, ਕਾਰਨ ਸੁੱਕ ਇਹ ਸੂਆ, ਇਹ ਸੁੱਕ, ਇਹ ਰਿਹਾ ਸੂਆ ਸੁੱਕ, ਸੂਆ ਰਿਹਾ, ਸੂਆ ਹੈ ਸੁੱਕ ਰਿਹਾ, ਸੁੱਕ ਹੈ, ਰਿਹਾ ਹੈ

- Bigrams: The sequences of two words in the context of an instance are known as the bigram features. The bigrams include the ambiguous target word along with all surrounding words in an instance. The bigram features are based on the window size. The default window size is 2 for selecting the bigrams from the context. While in the context, indicative words of

ambiguous target word may not occur consecutively. They may have multiple words between them. It is due to this reason diverse window size has applied to include the bigrams which have 3 or 4 words in between them. Table 5.6 shows the different bigram features selected from the instance having the target word ਸੂਆ.

The selection of the bigram features from the sense tagged dataset of the individual Punjabi ambiguous word has been defined in Algorithm 5.4. This algorithm traverses the sense files of the ambiguous word and generates the bigram features by processing each sentence of the individual sense file of the ambiguous word. In this research work, the parameter window size in algorithm 5.4 has been set to 2, 3 and 4 independently to select three different bigram features.

Algorithm 5.4: Bigram Selection

```
/* List of bigrams selected from all the sense files of the ambiguous word w */
Input:  w, window_size
Output:  bigram_list
for each sense file of the w do
    separate each sentence of the w
    generate the word token by splitting sentences at " "
    for each token of sentence do
        for window_size do
            bigram_list.add(token, token+window_size)
        end for
    end for
end for
```

Numerous choices have been made based on the position of the surrounding words (to the left or right of the target word) for n-grams features. However, the bigram features that consist of two words around the target word have been considered the most effective feature in determining the sense of the target word (Mohammad & Pedersen, 2004b). Therefore, the bigram Algorithm 5.4 with 2, 3 and 4 window size has been applied on the sense tagged dataset to 150 ambiguous Punjabi words to prepare their feature sets.

The topical and local selected features have explored the lexical relations among the words. To study the effect of grammatical patterns on the senses of the words syntactic features have been explored.

5.2.4 Syntactic Feature

The syntactic features explore the different structural information of the context of the Punjabi language. These features extract the patterns of grammatical classes between the target word and the neighbouring words. The syntactic features also study the relationship between the

head word and target word cues in a sentence. Syntactic features have local scope. The details of the selection of the syntactic feature namely, the part of speech (PoS), are as follows.

5.2.4.1 Part of Speech

The context words of the Punjabi language have been classified into four different word classes (PoS tags) namely noun, verb, adjective and adverb. The PoS tags of the neighbouring words of the target ambiguous word vary according to its sense and thus can be utilised to identify its sense. Consider the example instances in Table 5.7. The PoS tag represented with a forward slash has been assigned to each token in the sentence. It has been observed from Table 5.7 that two or three PoS tag sequences around the target word having different senses are discrete. While the words farther from target words have the least impact on the identification of its sense.

Table 5.7: PoS tags for all senses of word ਸੂਆ (TT: sūā)

Sense	**Example instance**
S1	ਇਸ/DEM, ਝਗੜੇ/NN, ਕਾਰਨ/PSP, ਮੁਲਜ਼ਮਾਂ/NN, ਨੇ/PSP, ਉਸ/PRP, ਦੇ/PSP, ਪਿਤਾ/NN, ਨੂੰ/PSP, **ਸੂਆ/NN**, ਮਾਰ/VM ਕੇ/VAUX ਮਾਰ/VM ਦਿੱਤਾ/VAUX TT: is jhagṛē kāran mulzamāṃ nē us dē pitā nūṃ sūā mār kē mār dittā T: Due to this fight culprits have killed his father using a knife
S2	ਜਿਆਦਾ/JJ, ਗਰਮੀ /NN, ਦੇ/PSP, ਕਾਰਨ/NN, ਇਹ/DET, **ਸੂਆ/NN**, ਸੁੱਕ/VM, ਰਿਹਾ/VAUX, ਹੈ/VAUX TT: jiādā garmī dē kāran ih sūā sukk rihā hai T: Due to hot summer this pond dries.
S3	ਸਾਡੀ/PRP, ਗਊ/NN, ਦਾ/PSP, ਇਹ/PRP, ਪਹਿਲਾ/QO, **ਸੂਆ/NN**, ਹੈ/VAUX TT: sāḍī gaū dā ih pahilā sūā hai T: This is the first baby of our cow
S4	ਦੁੱਧ/NN, ਦੇ/PSP, ਦੰਦਾਂ/NN, ਤੋਂ/PSP, ਬਾਅਦ/NST, ਬੱਚੇ/NN, ਦਾ/PSP**, ਸੂਆ/NN**, ਨਿਕਲ/VM, ਆਇਆ/VAUX, ਹੈ/VAUX TT: duddh dē dandāṃ tōṃ bāad baccē dā sūā nikal āiā hai T: After the milk teeth the child has a new incisor
S5	ਮੱਝ/NN, ਦੇ/PSP, **ਸੂਆ/NN**, ਲਗਾਉਣ/VM, ਲਗਿਆਂ/VAUX, ਡਾਕਟਰ/NN, ਨੇ/PSP, ਮੱਝ/NN, ਦੀਆਂ/PSP, ਪਿਛਲੀਆਂ/NN, ਲੱਤਾਂ /NN, ਬੰਨ੍ਹ/VM, ਦਿਤੀਆਂ/VAUX TT: majjh dē sūā lagāuṇ lagiāṃ ḍākṭar nē majjh dīāṃ pichlīāṃ lattāṃ bannh ditīāṃ T: Before giving injection to the buffalo the doctor has tied its back legs

The PoS tags have been assigned to a sense tagged dataset of 150 ambiguous Punjabi words using the Punjabi shallow parser. The word classes of neighbouring words of target ambiguous words have been generated by the intermediary stage (part of speech generator) of the Punjabi shallow parser. Five different PoS combinations have been explored and evaluated from the parsed Punjabi text to judge the right sense of an ambiguous word. The first PoS feature consists of PoS tags of the target word (PoS_w) and PoS tag of the word (PoS_{w+1}) immediate right to it, as shown in Table 5.8. The second PoS combination includes the PoS tag of the target word (PoS_w) and the PoS tags of the word (PoS_{w-1}) immediate left to it. The third PoS

combination has the PoS tags of two immediate neighbouring words (PoS_{w-1}, PoS_{w+1}) of the target word (PoS_w). The fourth PoS combination includes the PoS tags of two words (PoS_{w-2}, PoS_{w-1}) to the left and two words (PoS_{w+1}, PoS_{w+2}) to the right of the target word. The fifth combination contains the PoS tags of the target word (PoS_w), the PoS tag of one word (PoS_{w-1}) left to it and the PoS tag of one word (PoS_{w+1}) right to it. Table 5.8 shows the extracted syntactic features, i.e., PoS classes of five neighbouring words of the target word ਸੂਆ (TT: sūā) from an example instance.

Table 5.8:: PoS features of word ਸੂਆ (TT: sūā)

Sentence: ਜਿਆਦਾ\JJ,ਗਰਮੀ\NN, ਦੇ\PSP, ਕਾਰਨ\NN, ਇਹ\DET, **ਸੂਆ\NN**, ਸੁੱਕ\VM, ਰਿਹਾ\VAUX, ਹੈ\VAUX TT: jiādā, garmī, dē, kāran, ih, sūā, sukk, rihā, hai T: Due to hot summer this pond dries.		
POS1	PoS_w, PoS_{w+1}	ਸੂਆ\NN, ਸੁੱਕ\VM
POS2	PoS_{w-1} PoS_w	ਇਹ\DET, ਸੂਆ\NN
POS3	PoS_{w-1} PoS_{w+1}	ਇਹ\DET, ਸੁੱਕ\VM
POS4	$PoS_{w-2}PoS_{w-1}$ $PoS_w PoS_{w+1}$ PoS_{w+2}	ਕਾਰਨ\NN, ਇਹ\DET, ਸੂਆ\NN, ਸੁੱਕ\VM, ਰਿਹਾ\VAUX
POS5	$PoS_{w-1}PoS_w PoS_{w+1}$	ਇਹ\DET, ਸੂਆ\NN, ਸੁੱਕ\VM

The five different combinations of PoS tags of the Punjabi language constitute the Punjabi syntactic features. These features have been prepared using Algorithm 5.5. This algorithm firstly assigns the PoS tags to each word in the sense tagged dataset of ambiguous Punjabi word by calling the Punjabi shallow parser. Following that, the five combinations of PoS tags of target words have been selected from the PoS tagged sense files of the individual Punjabi ambiguous word. The ML classifiers have explored the selected syntactic features to perform the Punjabi WSD.

Algorithm 5.5 Selection of PoS combinations

```
/* List of PoS selected from all the sense files of the ambiguous word w */
Input: w
Output: POS_list[5]
for each sense file of the w do
    separate each sentence of the w
    for each sentence do
        sentence = call punjabi_shallow_parser_PoS_Tagger(sentence)
        generate the word token by splitting sentences at " "
        if w exists in tokens then
        POS_list[1].add ([PoSw PoSw+1]
        POS_list[2].add ( [PoSw-1 PoSw])
        POS_list[3].add ( [PoSw-1 PoSw+1])
        POS_list[4].add ( [PoSw-2PoSw-1 PoSw PoSw+1 PoSw+2])
        POS_list[5].add ([PoSw-1 PoSw PoSw+1])
        end if
    end for
end for
```

5.2.5 Semantic Features

Semantic features explore the relatedness among the words in the context. These are also known as distributional features. It has been observed that the meaning of a word is related to the distribution of the words around it (Jurafsky & Martin, 2009). The words distributed among the one sense of the ambiguous word are not related to the words distributed among the other sense of the word. The two different types of semantic/distributional features, namely the count based (count of neighbouring words) and prediction-based (neural network based), have explored in this research work. The details of count and prediction-based semantic features are as follows.

5.2.5.1 Count Based Distributional Features

The count-based distributional features are derived from the context of the sense tagged dataset of the Punjabi language. The context of the target word can be a word, a paragraph or a document. The count-based distributional model estimates the relationship between the target word and its context with the help of the co-occurrence matrix. The rows of the co-occurrence matrix are the unique words of the vocabulary, whereas the different contexts (word, paragraph or document) represent the columns of the matrix. The rows of the matrix show the relationship between the words, whereas the columns of the matrix represent the relation between the context. The word×document co-occurrence matrix describes the relationship between words and their respective senses files having instances of the target ambiguous words.

The word×document matrix consists of *N* rows of the unique words present in the sense files of the ambiguous word and *K* columns for the sense files of the ambiguous word. For instance, the word × document matrix *M* of the ambiguous word t_w having three different senses S_1, S_2 and S_3, is given in equation (5.1). The rows of matrix *M* are the *N* unique words, and the columns are the three different sense files of the ambiguous word t_w. The elements of the matrix *M* given in equation (5.1) are the frequency of the unique word w_1^f occurring in the respective sense file of the ambiguous word.

$$M = \begin{matrix} S_1 & S_2 & S_3 \\ w_1^{f^1} & w_1^{f^2} & w_1^{f^3} \\ w_2^{f^1} & w_2^{f^2} & w_2^{f^3} \\ \vdots & \vdots & \vdots \\ w_n^{f^1} & w_n^{f^2} & w_n^{f^3} \end{matrix} \quad \ldots (5.1)$$

Table 5.9 shows a few rows of the matrix *M* of the ambiguous word ਸੁਆ (TT: sūā,). The columns of Table 5.9 show the unique words selected from the five sense files of the ambiguous word

ਸੂਆ (TT: sūā,). The columns of Table 5.9 represent the sense files of the ambiguous word ਸੂਆ (TT: sūā,).

Table 5.9: word × document co-occurrence matrix of the ambiguous word ਸੂਆ (TT: sūā)

Words	Documents				
	S1	S2	S3	S4	S5
⋮					
ਝਗੜੇ (TT: jhagṛē, T: Fight)	3	0	0	0	0
ਗਰਮੀ (TT: garmī, T: Summer)	0	2	1	0	0
ਗਊ (TT: gaū,T: Cow)	0	0	2	0	0
ਦੰਦ (TT: dandāṃ, T: Teeth)	0	0	0	4	0
ਲੱਤਾਂ (TT: lattāṃ, T: Legs)	0	0	0	0	1
⋮					

To further estimate the effective co-relation among the different words, the weights have been assigned to the word relations using the term vector model (Manning, et al., 2008). The term vector model calculates the weights of the words using term frequency and inverse document frequency parameters.

The term frequency is estimated by counting the occurrences of a term/word in the document. It considers all terms/words of the documents equally important. But the document size is variable. Therefore, to appropriately distribute the weights of words occurring variably among the different documents, the length of the document is also considered. The weight of the word is inversely proportional to the length of the document. Equation (5.2) calculates the term frequency *tf(w)* of the word *w* in the document S_j. It is the ratio of the number of times word *(w)* that occur in the document S_j to the total number of words in the document S_j.

$$tf(w) = \frac{N_w^{S_j}}{N^{S_j}} \quad \ldots (5.2)$$

The inverse document frequency estimates the weights of the words that occur unevenly in the group of documents. For instance, a word w_i occurs in n documents, and w_j occurs in m documents. If $n>m$, then the word w_i is a very generic term while w_j is a very specific term. The specific term should have more weight than the generic term. Therefore, the inverse document frequency weight of a word is inversely proportional to the number of documents in which it appears. Equation (5.3) estimates the normalized *idf(w)* weights for the *w* word. It is the logarithm ratio of the total number of documents N_s to the number of documents in which the *w* word occurs N_w.

$$idf(w) = \log \frac{N_S}{N_w} \quad \ldots (5.3)$$

The *tf-idf(w)* weight for the individual word is estimated as the product of the *tf(w)* and *idf(w)* of the word. Equation (5.4) shows the *tf-idf(w)* weights of word *w* as the product of its frequency with its occurrence in the documents (senses).

$$tf - idf(w) = \frac{N_w^{s_j}}{N^{S_j}} \times log \frac{N_s}{w_t} \quad \ldots (5.4)$$

Finally, the matrix *M* in equation (5.1) has been modified with the TF-IDF weights of the individual word belonging to three different senses S_1, S_2 and S_3 of an ambiguous word t_w. The prepared word × document co-occurrence matrix has acted as a lookup table for the training and the test dataset to produce a numeric vector. The count based word×document matrix has been devised for the individual 150 ambiguous noun words of the Punjabi language and has been applied to build the supervised ML classifiers.

The limitation of the co-occurrence matrix is that they require huge memory computational resources. The size of the word vector selected from the co-occurrence matrix is dependent on the number of senses of the ambiguous words. These word vectors are specific for ambiguous word, and they cannot be generalized. The count-based techniques are unsupervised, that is, the co-occurrence relations have been created from the corpus by different frequency and probability-based methods. To address the issues of the count-based distributional features, the neural network-based word embedding features have explored.

5.2.5.2 Predictive or Word Embedding Features

The word vectors that have been created using Neural Network (NN) are called prediction-based or word embedding vectors. These word embedding vectors represent a concept or a word given in the vocabulary in the low dimensional continuous space. The embedding vectors of similar words are close to each other, while the embedding vectors of different words are placed far away from each other. The word embedding has a fixed dimension irrespective of the size of the vocabulary. The word embedding captures the useful regularities of a language, such as syntactic and semantic information. The embedding vector offset characterized the relationships of a language. The word embedding gives a distributional representation of a w_n word in the corpus as $w_n = R^d$ where *d* is the dimension of the vector. The NN predictive models estimate the d-dimensional vector offset directly as a supervised task.

The term "word embedding" has first coined by (Bengio, et al., 2003) who have obtained the values for the word embedding by training a neural language model. (Collobert & Weston,

2008) have demonstrated the application of the word embedding in many downstream NLP tasks, which has been prepared by multilayer NN. Word2Vec word embedding algorithm based on shallow NN by (Mikolov, et al., 2013) has changed the course of the NLP research. Pennington et al. (2014, GloVe) have developed another embedding model that has produced embedding representations from the statistical calculation on word counting.

In this research work, word embedding methods, namely Word2Vec, Global Vector (GloVe) and FastText have been applied to the Punjabi language text to produce a different set of word

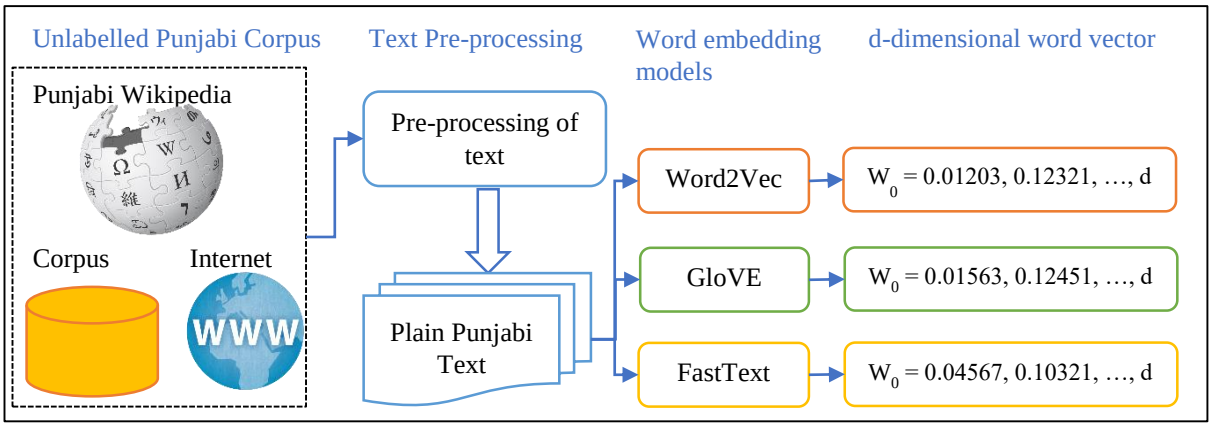


Figure 5.2: Process to generate the Punjabi word vectors

embedding features. The process of the creation of word embedding is shown in Figure 5.2. The word representations have been derived from the untagged corpus of the Punjabi language using shallow NN. The dataset has been pre-processed, and the shallow NN applied to it. The details of the process of the word embedding extraction models are as follows

(i) **Unlabelled Punjabi Corpus:** To prepare the Punjabi language word embedding features unlabelled dataset has been collected from the different online and offline resources of the Punjabi language. The major resource of the Punjabi language unlabelled corpus is the Punjabi language Wikipedia. The XML Wikipedia dump of the Punjabi language has been downloaded from the URL mentioned in Table 5.10. To collect more data for the Punjabi language, the written dataset of the Punjabi language has been obtained from the Evaluations and Language Resources Distribution Agency, Paris, France. This dataset is the collection of Punjabi news articles. (Goldhahn, et al., 2012) have collected the Punjabi language corpus from the different news websites of the Punjabi language. This Punjabi corpus is also included in this research work. The PoS tagged Punjabi language text prepared by the Indian Languages Corpora Initiative phase –II (ILCI Phase-II) project, initiated by the MeitY, Govt. of India, Jawaharlal Nehru University, New Delhi, has also been used to prepare the Punjabi language unlabelled corpus. Besides obtaining the Punjabi

corpus from Wikipedia and other Punjabi language resources for this research, we have also collected the Punjabi text from the internet or offline text documents. The details of the collected unlabelled Punjabi corpus are given in Table 5.10.

Table 5.10: List of the Punjabi unlabelled corpus

Name of Corpus	**Source/URL**	**Contents of dataset**	**Size of corpus in words**
WIKIPEDIA	https://dumps.wikimedia.org/backup-index.html downloaded on 23 March 2020	All subject areas	9.4 million
EMILLE Corpus	"The EMILLE/CIIL Corpus, ELRA catalogue (http://catalog.elra.info), ISLRN: 039-846-040-604-0, ELRA ID: ELRA-W0037".	News articles	1.4million
Leipzig Corpora Collection	https://wortschatz.uni-leipzig.de/en/download/panjabi	Web news articles	2 million
TDCIL POS tagged dataset	http://tdil-dc.in/	Agriculture, sports, religion, entertainment, literature	8 million
Manually prepared dataset	Punjabi newspapers, essays, stories and blogs from internet	Internet news, story, essay and blogs	0.47 million

(ii) **Pre-processing of the Punjabi language text:** The collected dataset of the Punjabi language from different resources, as shown in Table 5.10, includes the Punjabi text along with the additional meta-information about the text. However, to extract the plain text of the Punjabi language from these resources pre-processing task has been applied according to the type of the meta-information. Wiki Parser, a free and open-source software (FOSS) distributed under the MIT License by Peter (https://dizzylogic.com/wiki-parser/), has applied on the Punjabi language Wikipedia dump to removes the XML tags from the dump and to extract the plain Punjabi language text. A simple procedure in Python language has been developed to remove the XML tags from the EMILLE corpus, and the PoS tags from the TDCIL PoS tagged dataset to extract the plain text of the Punjabi language. The extracted plain text from the different language resources has been collected into a single plain text file of the Punjabi language. This file of the unlabelled text of the Punjabi language has been further pre-processed to remove the English language contents, numeric symbols and spelling aberrations of the Punjabi language words. This Punjabi language

unlabelled corpus contains 8,938,122 words. The pre-processed text has been used by the different word embedding model to derive the fixed dimensional word vectors.

(iii) **Word Embedding Models:** The functioning of individual Word2Vec, GloVe and FastText word representation models that have been used to produce the Punjabi language word embedding features are as follows.

- **Word2Vec** has been considered as the primary NN embedding model that has been developed by (Mikolov, et al., 2013) while working in Google. The Word2Vec model has been extensively applied by researchers to generate and tune word representations. (Mikolov, et al., 2013) has proposed two models of Word2Vec for building word representation, viz Continuous Bag of Words (CBOW) and Skip-Gram (SG) models. These techniques are based on two-layer NN architecture in which the hidden layer acts as the projection layer to reduce the computational requirements. The input to this shallow NN is the text corpus, and it produces a set of numeric vectors as the output. The detailed functioning of CBOW and SG models are as follows.

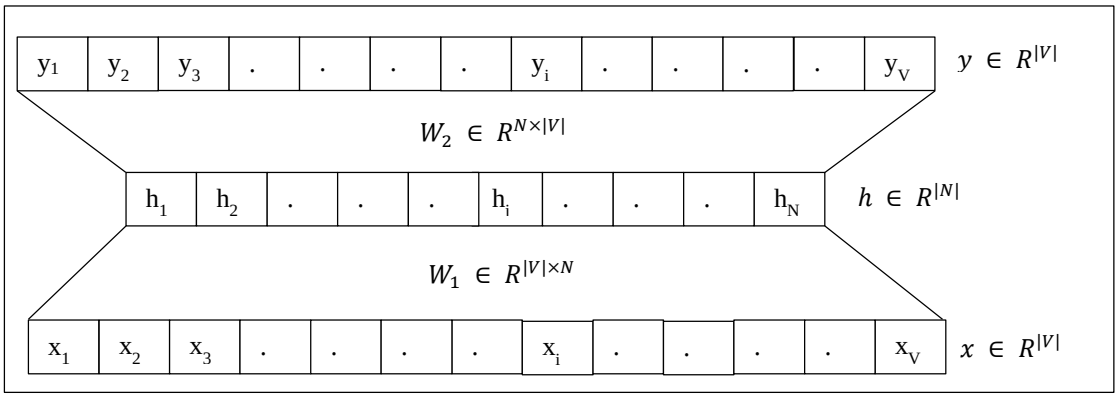


Figure 5.3: CBOW model with one-word context

- **CBOW** model predicts the centre word from the given context words. The context used by the CBOW model can be a single word or multiple words. The one-word CBOW model predicts one target word given one context word, which is similar to the bigram model. This model is a simple connected neural network having a single hidden layer of size *N*, input and output layers of vocabulary size *V* as shown in Figure 5.3. The input (*x*) is a one-hot encoded vector representation of *V* vocabulary words connected with *N* hidden nodes. The one-hot representation, also called one column vector representation is the binary representation of words. The size of one hot column vector is equal to the vocabulary size *V*. This binary vector assigns 1 for the target word in the vocabulary and 0 for the rest of the words in the vocabulary.

The weights between the input layer and the N-dimensional hidden layer (h) have been represented by a $V \times N$ matrix W_1. From the hidden layer to the output layer (y), there is a $N \times V$ weight matrix W_2. Each row of W_1 is N-dimensional vector representation of the associated input word. The one-hot representation of the word w selects the k^{th} row of the weight matrix W_1 ($V\times N$). The weights v_w of input word have linearly transferred from W_1 matrix to the N-dimensional hidden layer. The score of vocabulary words has been computed by taking the dot product of v_w (context word) weight of matrix W_1($V\times N$) and the v'_w (target word) weight of matrix W_2 ($N \times V$). Finally, the vocabulary scores of words have been converted into the probability distribution that is calculated by using a softmax classification model as given in equation (5.5).

$$p(w_j|w_I) = y_j = \frac{\exp({v'_{w_j}}^T v_{w_I})}{\sum_{j'=1}^{V} \exp({v'_{w_{j'}}}^T v_{w_I})} \quad \ldots (5.5)$$

Here v_w is the input vector and v'_w is the output vector representation of the word w selected from W_1 (input to hidden layer) and W_2 (hidden to output layer) weight matrix, respectively. The training objective of CBOW in equation (5.6) is to maximize the conditional probability of observing the actual output word w_j given the input context word w_I.

$$\max\left(p(w_j|w_I)\right) = \max(y_j) \quad \ldots (5.6)$$

The loss function is the cross-entropy measurement of the actual and predicted probabilistic distributions. To minimize the error between the projected probabilities of the target word and the output word, the weights of the matrix W_1 and W_2 have been adjusted using backpropagation. The stochastic gradient descent has been applied to adjust the weights of the matrices. After the model reaches the threshold limits the weight matrix

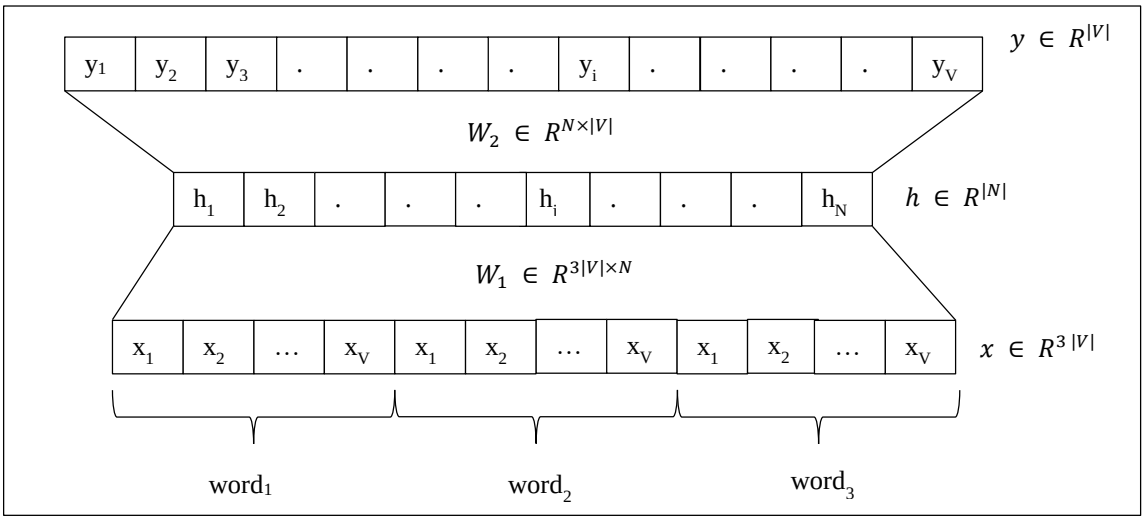


Figure 5.4:CBOW model with multi-word context

W_1 and W_2 act as the word embedding of the vocabulary. The two different weight matrices can be combined by applying different vector operations.

The second CBOW model takes into account the multiple words as the context to determine the target word and is called the multiword CBOW model, as shown in Figure 5.4. The CBOW computes the conditional probability of a target word given the context words surrounding it across a window of size *c*. In this model, the average of input vectors of context words is calculated before transferring the weights from the hidden layer to the output layer matrix. The loss function of the multiword CBOW is similar to the single word CBOW model, but it also includes *c* neighbouring words of the target word as shown in equation (5.7).

$$\frac{1}{T}\sum_{t=1}^{T}\log p(w_t|\sum_{-c\le j\le c, j\neq 0} w_{t+j}) \quad \ldots (5.7)$$

Here the conditional probability of the target word w_t is calculated using the same softmax model but considering context words *c* around it. The *c* is the window parameter that determines the words at the right and left sides of the target word. During backpropagation, the output hidden matrix weights have been updated similarly to the single word CBOW model. The update operation of hidden to input matrix weights is also similar, except that it has to be performed on every input context word.

- **SG model** predicts the context words surrounding the given target word instead of the

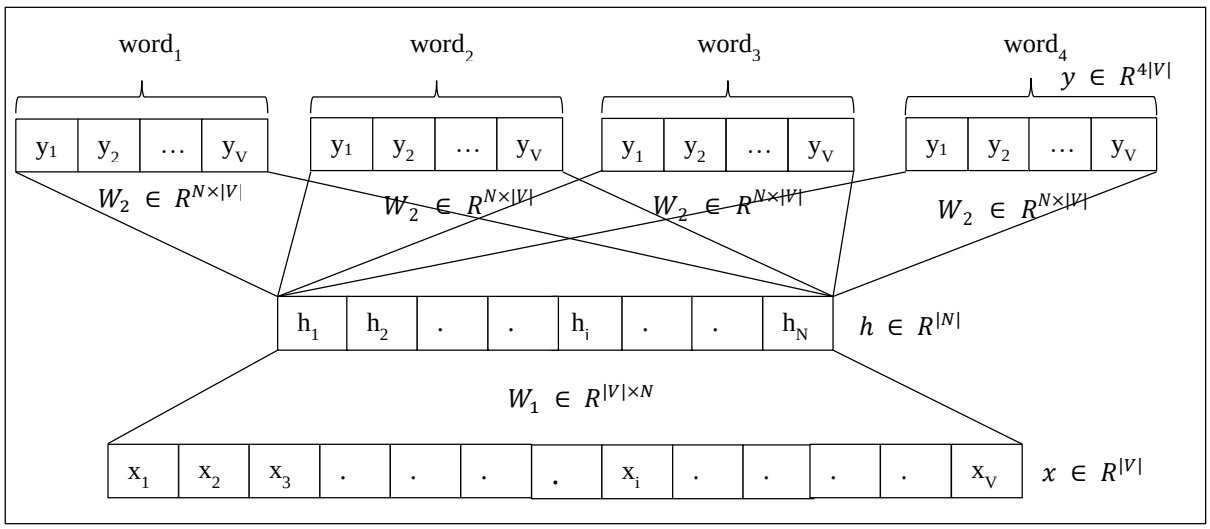


Figure 5.5: Skip Gram Word2Vec model

target word from context in the CBOW model. This Word2Vec model uses the target word as input and produces surrounding context words as output. The simple case when the SG model predicts one context word from one input word is similar to single word CBOW rules. However, for multiword context, it is based on the error estimations for the number

of surrounding words predicted as the context of the target word. The SG model that predicts the four context words that are two words each from the left and right side of the centre word, is depicted in Figure 5.5.

The input to the SG model is a one-hot encoded vector representation of the centre word connected with *N* hidden nodes. There is a $V \times N$ weight matrix W_1 between the input layer and the hidden layer. The linear activation function of the hidden layer simply copies the single row of W_1 corresponding to a single input word. The output layer represents the one-hot representation of *C* context words of the given input word in a window length of *c* (in Figure 5.7, c=2). From the hidden layer to the output layer, there are *C* (context words of the target word) $N \times V$ weight matrices W_2. The output layer represents the *C* multi nominal distributions instead of just one. The softmax function has been applied to obtain the posterior probability distribution of words. The training objective is to predict the surrounding words in a window length *c* of each word. The training objective function $J(\theta)$ in equation (5.8) maximizes the log probability of any context word w_{t+j} surrounding the centre word w_t and sum over all the possible context words. Then submit the objective function over all the possible input words of the vocabulary. The (θ) represents all the weight matrix parameters of the SG model and tries to learn these parameters.

$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} \log p(w_{t+j}|w_t) \quad \ldots (5.8)$$

The loss function of the SG is the sum of cross entropies of the multiple words. However, the softmax is computationally expensive to use, as computing the gradient has a complexity proportional to *c* predicted words each have vocabulary size of *V*. (Mikolov, et al., 2013) has proposed the application of negative sampling, and hierarchical softmax to overcome the computational limitations of the softmax function. Hierarchical softmax replaces the flat softmax layer and uses a binary tree to represent all words of the vocabulary. In the hierarchical softmax binary tree, there are w_V leaf words, *V-1* inner nodes and there exists a unique path from the root to the leaf node. The path from the root to the leaf node is used to estimate the probability of words represented at the leaf node. Hierarchical softmax evaluates at the most $\log_2|V|$ the depth of word in binary tree probabilities instead of computing *V* probabilities of a word to normalize its probability in the flat softmax. To further improve the encoding of more information in the tree frequency of the words in the corpus has been leveraged. (Mikolov, et al., 2013) has used a Huffman

tree, for their hierarchical softmax, that generates coding by assigning the fewer bits to more common symbols.

Negative sampling updates only a sample of the output vectors per iteration as compared to all the vocabulary vectors in the softmax. The random and true sample has been prepared by selecting words that co-occur and the words that do not co-occur respectively. The random corpus is larger than the true corpus. (Mikolov, et al., 2013) has put the target word in a positive sample and has tried to update it only.

The SG model represents rare words effectively, and even for a small amount of data, its performance is better. However, the CBOW model represents frequent words evidently and is faster. In unsupervised settings, the word embedding dimension is determined by the accuracy of prediction. As the embedding dimension increases, the accuracy of prediction also increases until it converges at some point, which is considered the optimal embedding dimension as it is the shortest without compromising accuracy. The Word2Vec models use context window and thus are not able to explore from the global text statistics. Therefore, the Word2Vec models are not able to learn from the repeated pattern of the text in the corpus.

- **Global Vectors (GloVe):** This representation model by (Pennington, et al., 2014) follows the hybrid approach in which the word vectors have trained using the co-occurrence ratios between two words in a context that are strongly connected to their meaning. The GloVe unsupervised learning algorithm developed at Stanford University has leveraged the global count statistics instead of the local information. The GloVe representation algorithm code and the pre-trained word vectors are also available at the Stanford University website (https://nlp.stanford.edu/projects/GloVe/).

  The starting point for preparing the GloVe representation is the estimation of the global information (computed from the entire corpus) about the co-occurrence between *i* and *j* words. The $X_{i,j}$ in equation (5.9) encodes the count information (PPMI, PMI or frequency) between *i* and *j* words in a context.

$$X_{i,j} = P(j|i) = \frac{X_{i,j}}{X_i} \quad \ldots (5.9)$$

  The objective of the GloVe representation is to produce the word vectors v_i and v_j for *i* and *j* words whose dot product is similar to the co-occurrence count of words *i* and *j* in the context. Equation (5.10) calculates the dot product of the word vectors v_i and v_j.

$$v_i^T v_j = \log P(j|i) = \log X_{i,j} - \log X_i \quad \ldots (5.10)$$

Equation (5.11) determines the dot product of the word vectors v_j and v_i for the word *j* and *i* in the co-occurrence matrix. In equation (5.11) the order of words has reversed from the order of word in equation (5.10).

$$v_j^T v_i = \log P(i|j) = \log X_{i,j} - \log X_j \quad \ldots (5.11)$$

After the addition of equation (5.10) and equation (5.11) we get the learnable parameter on the right hand side and the counts on the left hand side of the equation (5.12).

$$v_i^T v_j = \log X_{i,j} - \frac{1}{2}\log X_j - \frac{1}{2}\log X_i \quad \ldots (5.12)$$

The $\log X_i$ and $\log X_j$ values depend only on the count of words *i* and *j* in the co-occurrence word matrix. Thus equation (5.12) can be further simplified by adding biases to the right hand side to produce a new equation (5.13).

$$v_i^T v_j + b_i + b_j = \log X_{i,j} \quad \ldots (5.13)$$

The objective function for the GloVe word representations has been devised in equation (5.14) for all the word combinations occurring in the co-occurring matrix. This objective function has been represented into a least squares optimization problem.

$$min_{v_i v_j b_i b_j} \sum_{i,j} \left(v_i^T v_j + b_i + b_j - \log X_{i,j}\right)^2 \quad \ldots (5.14)$$

In equation (5.14), equal weights have been assigned to all the co-occurrences. The weights function $f(X_{ij})$ has been added in equation (5.14) to neutralize it. The weight function should have normalized the weights of rare and frequent words. The weight function for a pair of words $f(X_{ij})$ is defined in equation (5.15).

$$f(X_{ij}) = \begin{cases} (X_{ij}/X_{max})^{\alpha} & if\ X_{ij} < X_{max} \\ 1 & otherwise \end{cases} \quad \ldots (5.15)$$

The GloVe representation by (Pennington, et al., 2014) has adjusted the parameters at *max* =100 and α = ¾ values given in equation (5.15). The objective function after the addition of the weight function shown in equation (5.16) has been optimized using gradient descent such that word vector v_i and its context word v_j values are close to the co-occurring value of *i* word and *j* word in the co-occurring matrix $X_{i,j}$. The training algorithm repeat through all the elements of the co-occurrence matrix.

$$J = \sum_{i,j} f(X_{ij})\left(v_i^T v_j + b_i + b_j - \log X_{i,j}\right)^2 \quad \ldots (5.16)$$

The GloVe representations scale on vocabulary size. This is because the GloVe keeps a co-occurrence matrix for all word pairs and uses this matrix for training. The GloVe algorithm is usually slightly faster than the Word2Vec (Pennington, et al., 2014).

- **FastText:** This word representation model explores the idea of modular representation (Bojanowski, et al., 2017). The FastText modular embedding approach computes a vector for subword components per word. Later it computes the final word embedding by combining these subword embedding using a simple composition function. The FastText embedding learns subword vectors of a word using n-gram of characters. For example, the word "artefact" with *n*=3, has <ar, art, rte, tef, efa, fac, act, ct> FastText subword representation. The advantage of using the FastText sub word embedding is that it correctly captures the subword information, such as morphological variations. The FastText model do not consider the internal structure of the word. This character level representation enables the FastText representations to capture the meanings of suffixes and prefixes. In the FastText subword embedding each component in the vocabulary repeats more and thus it needs less training data. Nonetheless, embedding retrieval is slower because of the need for a composing function.

  The FastText algorithm has derived from the Word2Vec (SG) model with negative sampling. The objective function of the FastText is given in the equation (5.17) that determines the pairwise score between a word w_t and its context word w_c using negative log likelihood.

$$\sum_{t=1}^{T} \sum \log\left(1 + e^{-s(w_t, w_c)}\right) + \sum_{n \in N_{t,c}} \log\left(1 + e^{-s(w_t, w_c)}\right) \quad \ldots (5.17)$$

  To take into account the internal structure of the words, scoring function *s* has been modified such that it includes the set of n-grams appearing in word *w*. A vector representation has assigned to each n-gram of the word and the sum of these vector representations represent a word as given in equation (5.18)

$$s(w, c) = \sum_{g \in G_w} z_g^T v_c \quad \ldots (5.18)$$

  The word *w* is included in the set of its n-grams. The set of n-grams is thus a superset of the vocabulary. This model shares the representation of n-gram sets of words with other words, therefore, helps it to learn the representation for the rare words. The main difference between the Word2Vec and the FastText is that the FastText is a sub word model in which each word is subdivided into a series of n-grams. The FastText model bears the additional

complexity of splitting each word into its components, fetching their corresponding embedding vectors and compose them into the final word embedding.

(iv) **Generation of word embedding features:** The above-discussed word representation models have been applied to the unlabelled Punjabi text to create the Punjabi language word vectors. These models have been implemented using different tools, which are listed as follows.

a) The Generate Similar (Gensim), an open-source library, has implemented the Word2Vec models. The genism library handles the large text collections efficiently and provides tools to load different types of word embedding models (Rehurek & Sojka, 2010). Gensim is a Python-based library, and it requires NumPy and scipy libraries for installation on the machine. Gensim library provides the similarity measuring functions that can be used to evaluate the quality of the word embedding features extracted from the Punjabi language text.

b) The GloVe python library has been used to extract the GloVe word representations for the Punjabi Language from the unlabelled corpus.

c) The *fasttext* python module has been used to prepare the FastText Punjabi word vectors (Bojanowski, et al., 2017).

To create the Punjabi language word embedding features these libraries for the Word2Vec, GloVe and FastText have been applied individually to the Punjabi language unlabelled corpus. These NN models have defined hyperparameters that can be tweaked to extract the improved word representations. Table 5.11 shows the parameters, description and values of the Word2Vec model given in the gensim library.

Table 5.11: Hyper parameters for Word2Vec model

Hyper parameter	**Description**	**Values**
size	The Dimensionality of the word vectors	100
window	Number of words towards left and right of the target word	5
min_count	Minimum frequency of words in corpus to be considered	5
workers	Number of threads to train the model	4
sg	SG (0) or CBOW (1)	0 and 1
hs	Hierarchical softmax function	0
negative	If set to 0 no negative sampling else negative sampling specifies how many “noise words” should be drawn.	5
alpha	Leaning rate	0.01
iter	Epoch	20

The Word2Vec with these default value parameters has been trained on the unlabelled Punjabi language corpus.

The GloVe python library have assigned the default values for the hyper parameters such as no_of_parameter = 100 (dimension of word vector), learning_rate (gradient descent), cooccurence_matrix = 4 (number of words), epoch =10 and no_of_threads =10 while training of the GloVe model.

The FT word embedding model has used the same hyperparameter values as of the Word2Vec model shown in Table 5.11. In addition to that three additional parameters, namely min_n =3 (min length of char n-grams), max_n = 6 (max length of char n-grams), and bucket = 2000000 (number of buckets used for hashing n-grams), have applied.

5.2.5.3 Quality of The Punjabi Word Embedding Features

The prepared Word2Vec, GloVe and FastText Punjabi word embedding vectors have been evaluated by performing the similarity and analogy tests. These are known as the intrinsic operations that express the quality of the vectors. The embedding features have yielded numeric value vectors for the words of the Punjabi language corpus. The first subtask returns the most similar words of the asked target word. In the second subtask, analogy completion using the cosine similarity operation has been performed on word vectors to produces the most analogues word in the query. The results of the Punjabi language Word2Vec, GloVe and FastText word embedding features for similarity and analogy tests are as follows.

(i) **Most similar words:** In this task, the most related words have been determined from the Word2Vec (SG) word embedding features of the Punjabi language. Table 5.12 shows the ten most related words for the six different Punjabi ambiguous noun words returned by the Word2Vec (SG) word embedding vector representations.

Table 5.12: Most similar words extracted from Word2Vec (SG) word embedding vectors

Words	**ਫਲ** **TT: phal**	**ਹਾਰ** **TT: hār**	**ਜੰਗ** **TT: jaṅg**	**ਗੱਲ** **TT: gall**	**ਘਾਟ** **TT: ghāṭ**	**ਝਾੜ** **TT: jhāṛ**
Most similar words returned by Word2Vec Punjabi embedding representations	ਖਾਣਯੋਗ TT:khāṇyōg (0.708)	ਜਿੱਤ TT:jitt (0.641)	ਲੜਾਈ TT:laṛāī (0.754)	ਕਿੱਡੀ TT:kiḍḍī (0.663)	ਮੈਂਗਨੀਜ਼ TT:maiṅgnīz (0.597)	ਕੁਇੰਟਲ TT:kuiṇṭal (0.702)
	ਗਿਰੀਦਾਰ TT:girīdār (-0.705)	ਹਰਾਇਆ TT:harāiā (-0.628)	ਯੁੱਧ TT:yuddh (-0.706)	ਸਮਝਣੀ TT:samjhaṇī (-0.66)	ਕਮੀ TT:kamī (-0.594)	ਪਛੇਤੀ TT:pachētī (-0.642)
	ਫਲਾਂ TT:phalāṃ (-0.701)	ਹਾਰਨ TT:hāran (-0.623)	ਫ਼ੈਸਲਾਕੁੰਨ TT:phaaislākunn (-0.682)	ਉਜਰ TT:ujar (-0.657)	ਮੈਗਨੀਜ਼ TT:maignīz (-0.58)	ਪਛੇਤ TT:pachēt (-0.641)

	ਫਲਦਾਰ TT:phaldār (-0.699)	ਕਰਾਰੀ TT:karārī (-0.623)	ਜੰਗਾਂ TT:jaṅgāṃ (-0.667)	ਚਾਹਵਾਂਗਾ TT:cāhvāṅgā (-0.655)	ਸੋਕਿਆਂ TT:sōkiāṃ (-0.578)	ਪਕਾਵੇਂ TT:pakāvēṃ (-0.624)
	ਅਲੂਚਾ TT:alūcā (-0.689)	ਫਾਈਨਲ TT:phāīnal (-0.608)	ਜੰਗਾ TT:jaṅgā (-0.651)	ਸੁਣਨੀ TT:suṇnī (-0.655)	ਟੈਸਟੋਸਟੀਰੋਨ TT:ṭaisṭōsṭīrōn (-0.573)	ਬੇਮੌਸਮੀ TT:bēmausmī (-0.608)
	ਆੜੂ TT:āṛū (-0.687)	ਜਿੱਤੇਗੀ TT:jittēgī (-0.602)	ਸਭਰਾਵਾਂ TT:sabharāvāṃ (-0.636)	ਚਹੁੰਦੇ TT:cahundē (-0.654)	ਨੁਰੋਮਸਕੂਲਰ TT:nurōmsakūlar (0.572)	ਰਾਈਜ਼ੋਬੀਅਮ TT:rāīzōbīam (-0.604)
	ਗੁਠਲੀਦਾਰ TT:guṭhlīdār (-0.685)	ਵਾਕਓਵਰ TT:vākōvar (-0.602)	ਜਾਂਬਾਜ TT:jāmbāj (-0.632)	ਸੱਚੋ TT:saccō (-0.654)	ਗਰੱਭ TT:garabbh (-0.566)	ਨਿਸਰਣ TT:nisraṇ (-0.597)
	ਹਿੜ੍ਹਕਦਾਰ TT:hiṛhkadār (-0.682)	ਖਿਤਾਬੀ TT:khitābī (-0.6)	ਯੁੱਧਾਂ TT:yuddhāṃ (-0.629)	ਤੱਸਲੀ TT:tasslī (-0.654)	ਹਾਈਪੈਕਸਿਆ TT:hāīpauksiā (-0.562)	ਕਟਾਈਆਂ TT:kaṭāīāṃ (-0.596)
	ਅੰਜੀਰ TT:añjīr (-0.679)	ਤੋੜਵੀਂ TT:tōṛvīṃ (-0.595)	ਲੜਾਈਆਂ TT:laṛāīāṃ (-0.621)	ਲਫਜ਼ਾਂ TT:laphzāṃ (-0.651)	ਸੁੰਗੜਦੀਆਂ TT:suṅgaṛdīāṃ (0.559)	ਸੱਠੀ TT:saṭṭhī (-0.596)
	ਖਾਏਗਾ TT:khāēgā (-0.675)	ਹਮਵਤਨ TT:hamvatan (-0.595)	ਆਂਗਲ TT:āṅgal (-0.618)	ਗੁਜ਼ਾਰਿਸ਼ TT:guzārish (-0.651)	ਡੈਂਟਿਨ TT:ḍaiṇṭin (-0.557)	ਰਲਵੀਆਂ TT:ralvīāṃ (-0.596)

- Table 5.13 shows the ten most related words for six ambiguous Punjabi noun words returned by the GloVe word embedding vectors.

Table 5.13: Most similar words extracted from GloVe word embedding vectors

Words	**ਫਲ** **TT: phal**	**ਹਾਰ** **TT: hār**	**ਜੰਗ** **TT: jaṅg**	**ਗੱਲ** **TT: gall**	**ਘਾਟ** **TT: ghāṭ**	**ਝਾੜ** **TT: jhāṛ**
Most similar words returned by GloVe Punjabi embedding representations	ਸਬਜੀਆਂਸ਼੍ਰੇਣੀ TT:sabjīāṃsrēṇī (-0.94)	ਸਿੰਗਾਰ TT:siṅgār (-0.741)	ਲੜਾਈ TT: laṛāī (-0.79)	ਪੇਰੀਸ TT: pērīs (-0.846)	ਬਜ TT:baj (-0.801)	ਕੁਇੰਟਲ TT:kuiṇṭal (-0.794)
	ਹਿੜ੍ਹਕਦਾਰ TT:hiṛhkadār (-0.935)	ਹੀਰਿਆਂ TT: hīriāṃ (-0.736)	ਕੁੱਕੜਾਂ TT: kukkṛāṃ (-0.769)	ਕਰੀਏ TT: karīē (-0.821)	ਜ਼ਿੰਕ TT: jaiṅk (-0.791)	ਔਸਤਨ TT: austan (-0.682)
	ਖਾਏਗਾ TTkhāēgā (-0.921)	ਬਿਜਲੀਆਂ TT:bijlīāṃ (-0.733)	ਮੈਦਾਨੇ TT: maidānē (-0.765)	ਮੁੱਕਦੀ TT: mukkdī (-0.793)	ਸ਼ਮਸ਼ਾਨ TT: samsaān (-0.773)	ਲੀਟਰ TT: līṭar (-0.63)
	ਮਾਡੀਸਨ TTmāḍīsan (-0.912)	ਕਰਾਰੀ TT: karārī (-0.726)	ਲਿਓਯੇਂਗ TT: liōyēṅg (-0.744)	ਹੈਰਾਨੀ TT: hairānī (-0.771)	ਮੈਂਗਨੀਜ਼ TT: maiṅgnīz (-0.762)	ਔਸਤ TT: ausat (-0.627)
	ਗੁਠਲੀਦਾਰ TTguṭhlīdār (-0.88)	ਹੱਥੋਂ TT:hatthōṃ (-0.705)	ਮੈਦਾਨਿ TT: maidāni (-0.723)	ਕਹਾਗਾਂ TT: kahāgāṃ (-0.749)	ਅਹੀਰਤਲਾ TT: ahīrtalā (-0.742)	ਝੰਬ TT: jhamb (-0.609)
	ਕਾ TT:kā (-0.847)	ਸ਼ਿੰਗਾਰ TT:saiṅgār (-0.705)	ਕਾਰਗਿਲ TT: kārgil (-0.716)	ਆਖੀ TT: ākhī (-0.748)	ਉਤਾਰ TT: utār (-0.712)	ਵਚਨਬੱਧਤਾ TT:vacnabddhatā (-0.577)
	ਆਊਜ਼ TT: āūja꞉ (0.830)	ਕਾਂਸਤਾਂਤਨੋਪਲ TT:kāṃstāntnōpal (-0.655)	ਟਰੋਜਨ TT: ṭarōjan (-0.712)	ਸੁਣੀਂ TT: suṇīṃ (-0.739)	ਧੋਬੀ TT: dhōbī (-0.703)	ਪ੍ਰਤੀ TT: pratī (-0.573)

	ਫਲਸ਼੍ਰੇਣੀ TT:phalsarēṇī (-0.825)	ਲੜਾਈ TT: laṛāī (-0.624)	ਸੰਸਾਰ TT: saṃsār (-0.711)	ਕਹੀ TT: kahī (-0.733)	ਬਾਢ TT: bāḍh (-0.691)	ਹਫ਼ਤਾ TT: haftā (-0.568)
	ਫੁੱਲ TT: phull (-0.709)	ਕੁੱਕੜਾਂ TT: kukkṛāṃ (-0.623)	ਟੇਨਨਬਰਗ TT: ṭēnnabrag (-0.711)	ਇਹ TT: ih (-0.731)	ਸ਼ਮਸ਼ਾਨ TT: shamshān (-0.688)	ਵਿਕਣ TT: vikaṇ (-0.559)
	ਸਬਜੀਆਂ TT:sabjīāṃ (-0.703)	ਸਬਮਿਸਨ TT: sabmisan (-0.614)	ਯੁੱਧ TT: yuddh (-0.709)	ਬਾਤ TT: bāt (-0.731)	ਕਮੀ TT: kamī (-0.669)	ਕੈਲੋਰੀ TT: kailōrī (-0.558)

- Table 5.14 shows the ten most related words for six ambiguous Punjabi noun words returned by the FT word embedding features.

Table 5.14: : Most similar words extracted form Fatext word embedding vectors

Words	**ਫਲ** **TT: phal**	**ਹਾਰ** **TT: hār**	**ਜੰਗ** **TT: jaṅg**	**ਗੱਲ** **TT: gall**	**ਘਾਟ** **TT: ghāṭ**	**ਝਾੜ** **TT: jhāṛ**
Most similar words returned by Fasttext Punjabi embedding representations	ਗਿਰੀਦਾਰ TT:girīdār (0.691)	ਹਾਰੀ TT:hārī (0.744)	ਜੰਗਾ TT:jaṅgā (0.785)	ਗੱਲੀ TT:gallī (0.755)	ਘਾਟਾਂ TT:ghāṭāṃ (0.609)	ਝਾੜੋਂ TT:jhāṛōṃ (0.719)
	ਔਸ਼ਧ TT:ausa◌ dh (0.676)	ਹਾਰਨ TT:hāran (0.741)	ਲੜਾਈ TT:laṛāī (0.765)	ਗੱਲਬਤ TT:gallbat (0.742)	ਘਾਟਿਆਂ TT:ghāṭiāṃ (0.606)	ਝਾੜਾ TT:jhāṛā (0.718)
	ਗੁਠਲੀਦਾਰ TT:guṭhlīdār (0.673)	ਜਿੱਤ TT:jitt (0.717)	ਯੁੱਧ TT:yuddh (0.745)	ਗੱਲੋਂ TT:gallōṃ (0.737)	ਘਾਟੇਵੰਦਾ TT:ghāṭēvndā (0.605)	ਝਾੜਨ TT:jhāṛan (0.717)
	ਖਰਬੂਜ਼ਾ TT:kharbūzā (0.673)	ਹਾਰੇ TT:hārē (0.714)	ਜੰਗਜੂ TT:jaṅgjū (0.72)	ਗੱਲਤ TT:gallat (0.725)	ਸ਼ਮਸ਼ਾਨਘਾਟਾਂ TT:sa◌msaānghāṭāṃ (0.596)	ਝਾੜੀ TT:jhāṛī (0.714)
	ਖਰਬੂਜ਼ੇ TT:kharbūzē (0.668)	ਕਰਾਰੀ TT:karārī (0.705)	ਯੁੱਧਾ TT:yuddhā (0.715)	ਗੱਲ਼ TT:galla◌ (0.720)	ਘਾਟਗੇ TT:ghāṭgē (0.589)	ਝਾੜੂ TT:jhāṛū (0.685)
	ਫੁੱਲਗੋਭੀ TT:phullgōbhī (0.667)	ਜਿੱਤਦੀ TT:jittdī (0.695)	ਯੁੱਧਬੰਦੀ TT:yuddhbandī (0.709)	ਗੱਲਾ TT:gallā (0.715)	ਉਪਜਾਊਪੁਣੇ TT:upjāūpuṇē (0.585)	ਕੁਇੰਟਲ TT:kuiṇṭal (0.676)
	ਫਲੀਦਾਰ TT:phalīdār (0.665)	ਹਾਰਾਂ TT:hārāṃ (0.682)	ਯੁੱਧਖੇਤਰ TT:yuddhkhētar (0.7)	ਗੱਲਾਂ TT:gallāṃ (0.704)	ਸ਼ਮਸ਼ਾਨਘਾਟ TT:sa◌msaānghāṭ (0.584)	ਝਾੜੀਨੁਮਾ TT:jhāṛīnumā (0.657)
	ਪੱਤਝੜੀ TT:pattjhaṛī (0.662)	ਜਿੱਤੀ TT:jittī (0.679)	ਜੰਗਬਾਜ਼ TT:jaṅgbāz (0.699)	ਠਰੀ TT:ṭharī (0.701)	ਬਾਬੂਘਾਟ TT:bābūghāṭ (0.57)	ਝਾੜੀਦਾਰ TT:jhāṛīdār (0.654)
	ਔਸ਼ਧੀਏ TT:aus◌dhīē(0.661)	ਜਿੱਤਾ TT:jittā (0.676)	ਲੜਾਈਆ TT:laṛāīā (0.698)	ਗੱਲੇ TT:gallē (0.697)	ਕਮੀ TT:kamī (0.562)	ਝਾੜਦਾ TT:jhāṛdā (0.64)
	ਫਲੀ TT:phalī (0.658)	ਹਰਾਇਆ TT:harāiā (0.669)	ਜੰਗੇ TT:jaṅgē (0.69)	ਗੱਲੀਂ TT:gallīṃ (0.696)	ਖਾਰੇਪਨ TT:khārēpan (0.561)	ਪਛੇਤੀ TT:pachētī (0.633)

(ii) **Analogies completion task:** In this task, the Punjabi word embedding features have identified the word vector with maximum cosine similarity for an analogy. For example an

analogy in the form a:b::c:? has been completed by calculating the cosine similarity among the word embedding vectors of words w_a, w_b, and w_c. Equation (5.19) calculates the word vector for analogy completion task.

$$d = argmax \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||} \quad \ldots (5.19)$$

An analogy can be interpreted like queen – king = actress – actor. Thus to identify the missing value of analogy, the word vector of the words in analogy can be written as $w_d = w_b - w_a + w_c$. The w_d has been identified as cosine similarity that maximizes the normalized dot-product between the two word vectors. Table 5.15 shows the results of the three different word embedding features used for the analogy completion task.

Table 5.15: Comparison of Analogy completion task

Relationship type Man Woman	**Manual Analysis**	**Word2Vec**	**GloVe**	**FT**
ਰਾਜਾ-ਰਾਣੀ :: ? – ਕੁੜੀ TT: rājā-rāṇī :: ? - kuṛī	ਮੁੰਡਾ TT: muṇḍā	ਮੁੰਡੇ TT: muṇḍē (0.603) **ਮੁੰਡਾ TT: muṇḍā (0.52)**	ਮੁੰਡੇ TT: muṇḍē (0.235)	ਮੁੰਡੇ TT: muṇḍē (0.683) ਕੁੜੀਉ TT: kuṛīu (0.671)
ਰਾਜਾ-ਰਾਣੀ::? –ਭੈਣ TT: rājā-rāṇī::?-bhaiṇ	ਭਰਾ TT: bharā	**ਭਰਾ TT: bharā (0.667)** ਪੁੱਤਰ TT: puttar (0.553)	ਛੋਟੀ TT: chōṭī (0.076)	**ਭਰਾ TT: bharā (0.661)** ਪੁੱਤਰ TT: puttar (0.599)
ਰਾਜਾ-ਰਾਣੀ::?-ਇਸਤਰੀ TT: rājā-rāṇī::?-istarī	ਪੁਰਸ਼ TT: purash	ਈਡੀਪਸ TT: īḍīpas(0.475) ਪੁਰਸ TT: purash (0.466)	ਕੈਮਾਂਤਰੀ TT: kaumāntrī (0.467)	ਇਸਤਰੀਆਂ TT: istarīāṃ (0.594) ਸਤਰੀ TT: satarī (0.542)
ਰਾਜਾ-ਰਾਣੀ ::?- ਅਭਿਨੇਤਰੀ TT: rājā-rāṇī::?-abhinētrī	ਅਭਿਨੇਤਾ TT: abhinētā	**ਐਕਟਰ TT: aikaṭar** (0.575) **ਅਦਾਕਾਰਾ TT: adākārā** (0.575)	**ਅਦਾਕਾਰਾ TT: adākārā** (0.273)	**ਅਦਾਕਾਰ TT: adākār (0.684)** ਅਭਿਨੇਤਾ TT: abhinētā (0.677)
ਭਾਰਤ-ਦਿਲੀ :: ? – ਇਸਲਾਮਾਬਾਦ TT: bhārat-dilī :: ? - islāmābād	ਪਾਕਿਸਤਾਨ TT:pākistān	**ਪਾਕਿਸਤਾਨ TT: pākistān** (0.562) ਅਫ਼ਗ਼ਾਨਿਸਤਾਨ TT: afġānistān (0.54)	ਚਾਂਦਨੀ TT: cāndnī (0.038)	ਤਜ਼ਾਕਿਸਤਾਨ TT: tajaākistān (0.684) ਤਜਾਕਿਸਤਾਨ TT: tajākistān (0.681)
ਭਾਰਤ-ਦਿਲੀ :: ? – ਢਾਕਾ TT: bhārat-dilī :: ? - ḍhākā	ਬੰਗਲਾਦੇਸ਼ TT:baṅglādēsa◌	**ਬੰਗਲਾਦੇਸ਼** **TT:baṅglādēsa◌** (0.649) ਢਾਕਾ (0.528)	ਮਿਲੀਆ TT: milīā (0.378)	**ਬੰਗਲਾਦੇਸ਼** **TT:baṅglādēsa◌** (0.606) ਸੀਭਾਰਤ (0.605)
ਭਾਰਤ-ਦਿਲੀ :: ? – ਕਾਬੁਲ TT: bhārat-dilī :: ? - kābul	ਅਫ਼ਗ਼ਾਨਿਸਤਾਨ TT: afġānistān	ਅਫ਼ਗ਼ਾਨਿਸਤਾਨ TT: afġānistān (0.568) ਤਾਲਿਬਾਨ TT: tālibān (0.521)	ਚਾਂਦਨੀ TT: cāndnī (0.377)	ਕਾਬੁਲੀ TT: kābulī (0.674) ਅਫਗਾਨਸਤਾਨ TT: aphgānsatān (0.634)
ਭਾਰਤ-ਦਿੱਲੀ :: ? – ਬਗ਼ਦਾਦ TT: bhārat-dillī :: ? - bag◌dād	ਇਰਾਕ TT: irāk	ਖ਼ੁਰਾਸਾਨ TT: khaurāsān (0.583) **ਇਰਾਕ TT: irāk** (0.546)	ਚਾਂਦਨੀ TT: cāndnī (0.423)	ਬਗ਼ਦਾਦੀ TT: bag◌dādī (0.673) **ਇਰਾਕ TT: irāk** (0.601)
ਭਾਰਤ-ਦਿਲੀ :: ? – ਮਾਸਕੋ TT: bhārat-dilī :: ? - māskō	ਰੂਸ TT: rūs	**ਰੂਸ TT: rūs** (0.606) ਬੁਲਗਾਰੀਆ TT: bulgārīā (0.573)	ਪੀਟਰਸਬਰਗ TT: pīṭrasbarag (0.292)	**ਰੂਸ TT: rūs** (0.633) ਬੁਲਗਾਰੀਆ TT: bulgārīā (0.617)

ਪੰਜਾਬ – ਚੰਡੀਗੜ੍ਹ :: ?- ਜੈਪੁਰ TT: pañjāb – caṇḍīgaṛh :: ?- jaipur	ਰਾਜਸਥਾਨ TT: rājsathān	**ਰਾਜਸਥਾਨ TT: rājsathān** (0.51) ਬੀਕਾਨੇਰ TT: bīkānēr (0.48)	ਜੰਕਸ਼ਨ TT: jaṅksa◌ॅn (0.334)	ਪੰਜਾਬ TT: pañjāb (0.596) ਸੀਪੰਜਾਬ TT: sīpañjāb (0.59)
ਪੰਜਾਬ – ਚੰਡੀਗੜ੍ਹ :: ?- ਪਟਨਾ TT: pañjāb – caṇḍīgaṛh :: ?- paṭnā	ਬਿਹਾਰ TT: bihār	**ਬਿਹਾਰ TT: bihār** (0.586) ਪੇਰੀਆਰ TT: pērīār (0.47)	ਫੋਰਟ TT: phōraṭ (0.363)	ਸੀਪੰਜਾਬ TT: sīpañjāb (0.616) **ਬਿਹਾਰ TT: bihār (0.602)**

Three different analogy relations have experimented and results are shown in Table 5.15. The first four rows represent the gender relation, the next five are the country capitals, and the last two are the state capital relations. To evaluate the word embedding features quality, the results of the Word2Vec, GloVe and FastText word embedding features have been compared with the manually determined answers to the analogies. The bold values in the Word2Vec, GloVe and FastText columns show that they match with the manually solved analogy task. It has been observed from Table 5.15 that the Word2Vec (SG) word embedding features have yielded the most suitable analogy words when compared with gold standard manual results. The FastText features have produces the second best, and the GloVe embedding features have returned the least suitable results. The evaluations of the word embedding features on the similar word and the analogy completion tasks indicate that the Punjabi word vectors generated by Word2Vec, GloVe and FastText models are rich, informative and can be further employed for the WSD of the Punjabi language.

The different topical, local, syntactic and semantic features have been selected from the sense tagged dataset and the unlabelled corpus of the Punjabi language. The individual selected features have been evaluated using the ML and deep learning classifiers.

Chapter Summary

The features are crucial for the training of the supervised ML classifiers and performing the Punjabi WSD. The context of the ambiguous words provides the necessary clues about the meaning of the word. The context has been transformed into a structured format, and undiscriminating stop words have been removed. This context has been used to extract the features of ambiguous words.

Different topical, local syntactic and semantical features have been selected in this chapter using the different selection algorithms. The baseline MFS feature provides the sense skewness information of the sense tagged dataset of the Punjabi language. The two topical features (BoW and co-occurrence) and two local features (collocations and bigram) have been selected from the sense tagged Punjabi dataset. The selection of different patterns of PoS features from the PoS tagged sense tagged Punjabi dataset forms the syntactic features. The count-based and prediction-based embedding features are the two semantic features used for the Punjabi WSD. The count-based features have been selected from the co-occurring matrix namely, word×document. The prediction-based embedding features have been derived from the unlabelled corpus of the Punjabi language. The Word2Vec, GloVe, and FastText three different NN based models have trained on the unlabelled corpus of the Punjabi language to generate the three different word representations. The quality of these word representations has been determined by selecting the most similar words and analogy completion task.

The topical, local, syntactic and count-based semantic features have been used by different ML classifiers for training. The word predictions based embedding features have facilitated the deep learning machine learning classifiers to perform the Punjabi WSD. Nevertheless, the word embedding features have also experimented on the ML classifiers. The implementation of the Punjabi WSD architecture using different features set has been discussed in the next chapter of the thesis.

Chapter 6 : Implementation of the Punjabi WSD

In this chapter the implementation of the Punjabi WSD has been discussed. The Punjabi WSD is based on the classification phase of the Punjabi WSD architecture. The classification phase of the Punjabi WSD architecture is a two-step process. In the first step, the training of the Machine Learning (ML) and deep learning classifiers have been performed by using the topical, local, syntactic and semantic features. The topical, local and syntactic features are the collection of strings values. These string features have been transformed into the real value vectors such that the ML classifiers can handle them. The two semantic features (count-based and prediction-based) generate the real value vectors for the individual Punjabi word. These two semantic feature vectors have been processed using concatenation, average, fractional and exponential vector combination approaches to train the ML classifiers. The deep learning classifiers have been directly trained on the prediction-based word embedding features and do not require any intermediator processing. The different hyperparameters, libraries and application tools have been used in the training of the ML and deep learning classifiers and discussed in this chapter.

In the second step, the trained classifiers predict the most suitable sense of the ambiguous word. The Punjabi WSD system receives a Punjabi instance as an input. It pre-processes, tokenizes and search the generated tokens in the Punjabi dataset. If it found the ambiguous word from the Punjabi dataset, it predicts the sense of the ambiguous word using the trained machine learning and deep learning models. The evaluation of the explicit Punjabi WSD is based on the implementation and training of the supervised ML and DL classifiers.

The data structures employed and the output details of the Punjabi WSD have been deliberated in this chapter. A web interface has been developed that integrates the different components of the Punjabi WSD.

6.1 Implementation of Classification Task for Punjabi WSD System

The classification task is the last but the most essential phase of the Punjabi WSD architecture. In this work, the ML supervised classifiers have trained to perform the sense classification of an ambiguous word. Figure 6.1 illustrates the working of the ML supervised classifiers to perform the classification task. The classification task is defined as a two-step process. In the first step, the classification algorithm builds the classifier by analysing a training set that consists of tuples and the associated class labels. In the second step, the trained model predicts

the sense of the target ambiguous word in an instance of the test dataset. The test dataset is independent of the training dataset, which means that the test dataset has not been used in the training of the classifier.

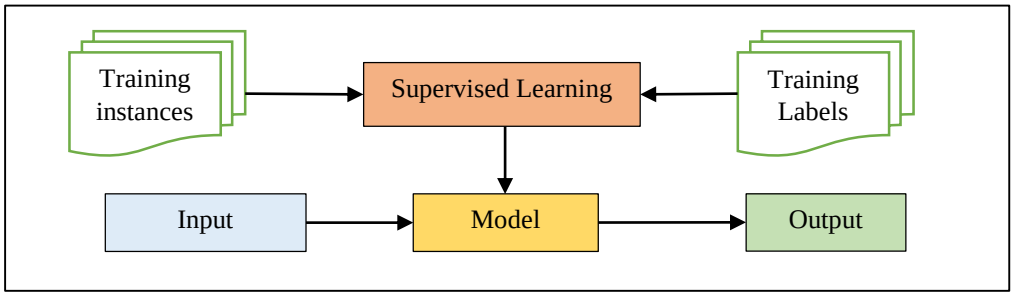


Figure 6.1:Supervised Learning

The dataset for the ML classifiers should consist of numeric values. However, the feature vectors selected from the Punjabi sense tagged dataset contains the strings values. To convert, the string feature sets into numeric values, the transformation process has been applied. The transformation is indispensable for the ML algorithms because they need numbers (in vectors) instead of words to perform the linear algebra operations. The details of the string to vector transformation process are as follows.

6.1.1 String to Vector Transformations

The topical, local and syntactic features selected from the Punjabi language sense tagged dataset consist of the combinations of the context words. To convert these context word features into numeric values, the transformation algorithms for topical (Bag of Words (BoW), co-occurrence), local (collocations and bigram) and syntactic (Part of Speech (PoS)) feature sets have been devised and are discussed as follows.

(i) **BoW string to numeric vector transformation:** The topical BoW feature vector consists of unique words present in a sense tagged dataset of an ambiguous Punjabi noun word and their corresponding frequencies in the dataset. Algorithm 6.1 has been applied to the string values of the BoW feature set of an ambiguous word to generate the numeric values for its training dataset. Algorithm 6.1 computes the frequency-based weights of each word present

Algorithm 6.1: Transformations of strings to binary vectors of BoW and co-occurrence string vectors

Input: *bow_ list*, *w* // BoW feature vectors, ambiguous word
Output: word_vector[dataset.size()][*bow_ list*.size()+1] //Binary training datasets

word_vector [dataset.size()][*bow_ list*.size()+1] = 0
/* Count BoW feature words in the dataset of the *w**/
for each sense file of the *w* **do**
  for each sentence **do**
    generate the *token* by splitting sentence at " "
    for each *token* of sentence **do**
      total_word_count[*token*] += 1
    end for
  end for
end for
/* Calculate the count of the words per sense file*/
for each sense file of the *w* **do**
  for each sentence **do**
    generate the *token* by splitting sentences at " "
    for each *token* of sentence **do**
      count_word[*token*] += 1
    end for
  end for
/* Calculate the weight of the word */
  for each sentence **do**
    generate the *word* by splitting sentences at " "
    for each *word* of sentence **do**
    for each *token* in *bow_ list* **do**
      if (*token* exist in *bow_ list***) then**
        word_vector[sentence][*bow_ list* (word)]=$\frac{count_word[token]}{total_word_count[token]}$
      end if
    end for
    end for
    word_vector[sentence][*bow_ list*(word)+1] = S_{label} //Sense label of the sentence
  end for
  count_word.*clear*()
end for

in a sense tagged dataset. According to Algorithm 6.1, the individual sense files of the ambiguous word have been traversed. Each sentence of the sense file is tokenised. The first loop in Algorithm 6.1 estimates the count of the discrete words of the sense tagged dataset. The second loop in Algorithm 6.1 determines the count of the tokens in the respective sense file. Each token in the sentence has been compared with the BoW feature of the ambiguous word. During this comparison, if the word exists in the BoW feature, the weight for the word has been assigned to it. The weight is the ratio of occurrence of the word in the specific sense file to the occurrence of that word in the whole sense tagged dataset. If no match between the sentence token and the BoW feature has been found, then 0 value is assigned to it. This comparison results in a floating value tuple of length equal to the size of the BoW feature. The dimension of each instance vector is equal to the length of the BoW vector.

Finally, Algorithm 6.1 generates the weighted numeric valued training dataset for an ambiguous word. This weighted dataset comprises tuples for each instance of the sense tagged dataset along with their labels. This process has been applied to all 150 ambiguous words of the Punjabi language. Conversely, this string to vector transformation yields a highly sparse matrix of the training dataset.

(ii) **Co-occurrence string to vector transformation:** Co-occurrence features consist of highly prominent words selected from the different sense files of the ambiguous word. The co-occurrence features select the words having a high frequency in the dataset of the ambiguous word. Algorithm 6.1 has also been applied to the co-occurrence string vector and the training dataset to generate the numeric training dataset for the ambiguous word. It computes the weights of the tokens in an instance by comparing them with the co-occurrence feature vector. The generated numeric training data consist of a weighted tuple for every instance of the sense tagged dataset of the ambiguous word. The dimension of each tuple is equal to the size of the co-occurrence feature. This transformation of the string dataset into numeric values using the co-occurrence string feature has been performed on the 150 ambiguous Punjabi noun words. Nonetheless, the size of the co-occurrence vector is smaller than the BoW vector, but it has yielded a highly sparse matrix of the training dataset than the BoW sparse matrix of the training dataset.

(iii) **Collocation to binary vector transformation:** There are two different types of collocation features selected for the individual target ambiguous word. The single word collocation has collected one word left and one word right to the target ambiguous word. The double word collocation has collected the two words left and two words right of the target word. The string collocation feature vectors have been applied to the individual instances of the sense files of the target word to produce the binary training dataset. Algorithm 6.2 describes the complete process of transformation of the string sense tagged dataset into the binary training dataset.

Algorithm 6.2 traverses each sense file of the sense tagged dataset. It collects the collocation combinations (single and double words) for an individual instance of the sense tagged dataset. Algorithm 6.2 compares the collected collocation from the sentences of sense tagged dataset with the collocation vector. Firstly, it compares a single word collocation vector with the individual token of the instance of the sense tagged dataset. The presence of collocation in a sentence is marked with 1, while the absence is marked with 0. For each instance, a binary vector of size equal to the size of the collocation vector of the ambiguous word has been created. This process has been applied to all the instances of the

sense tagged dataset of the ambiguous word, and it has yielded a binary training dataset. Secondly, this process has been applied for the double word collocation feature vector to produce the binary training dataset. The training matrices prepared during this process are also highly sparse.

Algorithm 6.2: Transformations of strings to binary vectors using Collocation word vectors

```
Input: coll1_ list, coll2_ list, w                    // Collocatiosn feature vectors, ambiguous word
Output: word_vector[dataset.size()][coll1_ list.size()+1],
 word_vector[dataset.size()][coll2_ list.size()+1]      //Binary training datasets

word_vector[dataset.size()][coll1_ list.size()+1] = 0
word_vector [dataset.size()][coll2_ list.size()+1] = 0
for each sense file of the w do
    for each sentence do                          // Collect collocations from sense tagged dataset
        generate the token by splitting sentence at " "
        if w in tokens then
            coll1 = token loc[w]-1, w, token loc[w]+1
            coll2 = token loc[w]-2, token loc[w]-1, w, token loc[w]+1, token loc[w]+2
        end if
/* Compare collocations with single word collocation vector to yield binary vectors*/
        for each token in coll1_ list do
            if (coll1 == token) then
                word_vector[sentence][coll1_ list (token)] = 1
            end if
        end for
        word_vector[sentence][ coll1_ list.size()+1]= Slabel  //Sense label of the sentence
/* Compare collocations with double word collocation vector to yield binary vectors*/
        for each token in coll2_ list do
            if (coll2 == token) then
                word_vector[sentence][ coll2_ list(token)] = 1
            end if
        end for
        word_vector[sentence][ coll2_ list.size()+1]= Slabel  //Sense label of the sentence
    end for
end for
```

(iv) **Bi-gram to numeric vectors transformation:** Three different bigram features have been used to prepare the three binary training datasets of an individual ambiguous word. Algorithm 6.3 describes the complete process of transformation of a string dataset into a binary training dataset using a bigram feature vector with window size 2. Algorithm 6.3 collects the binary combinations from each instance of the sense tagged dataset based on the window size 2, 3, and 4. It compares the tokens from the bigram feature with the bigram token of an individual instance of the sense tagged dataset of an ambiguous word. If any bigram from the feature set is present in the bigram selected from the sentence, it is set to 1, else; it is set to 0. The length of each tuple is equal to the length of the bigram feature. Similarly, the bigram feature vectors with window size 3 and 4 are used to transform the string vectors to binary vectors using Algorithm 6.3. Three different bigram features have

produced three different binary training datasets on which the ML classifiers have been trained.

Algorithm 6.3: Transformations of strings to binary vectors using Bigram word vectors

```
Input: bigram1_list, w                                   //Bigram feature vector, ambiguous word
Output: word_vector[dataset.size()][bigram1_list.size()+1] //Binary training dataset
word_vector[dataset.size()][bigram1_list.size()+1]=0
/*Collect the bigram combinations from each instance of the sense tagged dataset*/
for each sense file of the w do
    for each sentence do
        generate the tokens by splitting sentence at " "
        for each token of sentence do
        for window size do
            bigram1.add(token, token+window_size)     //window size used 2,3and 4
         end for
         end for
/*Compare the collected bigrams with bigram feature set*/
         for each token in bigram1 do
             if (token exist in bigram1_list)
                word_vector[sentence][bigram1_list (token)]=1
             end if
         end for
         word_vector[sentence][bigram1_list.size()+1]= Slabel //Sense label of the sentence
    end for
end for
```

(v) **PoS string to binary vector transformation**: The five different patterns ((PoS_w, PoS_{w+1}), (PoS_{w-1} PoS_w), (PoS_{w-1} PoS_{w+1}), (PoS_{w-2} PoS_{w-1} PoS_w PoS_{w+1} PoS_{w+2}), (PoS_{w-1} PoS_w PoS_{w+1})) of the PoS tags have been extracted from the PoS tagged dataset of an ambiguous word. Each PoS pattern forms a distinct feature vector. The dimension of each PoS vector is distinct and variable. Five different training datasets have been generated for these five different PoS feature vectors. Algorithm 6.4 has been applied to transform the PoS tagged

Algorithm6.4: Transformation of strings to binary vectors using PoS vectors

```
Input: PoS1_list, w                                      //PoS feature vector, ambiguous word
Output: word_vector[dataset.size()][POS1_list.size()+1] //Binary training dataset
 word_vector[dataset.size()][POS1_list.size()] = 0
/*Select the pattern from each sentences of the sense tagged dataset*/
for each sense file of the w do
    for each sentence do
        generate the tokens by splitting sentence at " "
        if w in tokens then
            PoS1 = token loc[w]-1, w, token loc[w]+1
        end if
/*Compare the selected pattern with PoS feature vector*/
        for each token in PoS1_list do
            if (PoS1 == token) then
               word_vector[sentence][ PoS1_list (token)] = 1
            end if
        end for
        word_vector[sentence][ PoS1_list.size()+1]= Slabel //Sense label of the sentence
    end for
end for
```

dataset into the binary value training dataset using the PoS vector (PoS_{w-1} PoS, PoS_{w+1}). It extracts the PoS (PoS_{w-1} PoS_w PoS_{w+1}) pattern from the individual instance of the sense tagged dataset of an ambiguous word. The selected patterns have been compared with the PoS feature vector. During the comparison, if the PoS vector value exists in the instance, the corresponding value in the vector tuple is set to 1. If no PoS vector value is found in the instance, then the value is set to 0. This comparison generates a binary tuple of the size equal to the size of the PoS feature vector.

After the execution of Algorithm 6.4, the training dataset of size equal to the number of instances in the PoS tagged dataset has been generated. Similarly, the other four PoS vectors have been compared with the PoS tagged dataset of the individual ambiguous word to create the binary vector training dataset.

The string to vector algorithms have been applied for the topical, local and syntactic features of the 150 ambiguous Punjabi words. However, the semantic features consist of the numeric values for the Punjabi language words selected from the sense tagged and unlabelled datasets. Therefore, the semantic features have been processed by applying the different combination approaches to generate the numeric vectors. The details of the feature vector creation using combination strategies are discussed as follows.

(vi) **Count-based semantic word vectors:** The co-occurrence word×document matrix has been prepared and explored in this work. The process of creation of word×document matrix has been discussed in chapter 5 of this thesis. The word vectors of word×document co-occurrence matrix have prepared from the sense tagged dataset. The ambiguous Punjabi word training dataset has been transformed into a numeric values dataset by two different methods. Firstly, the word vectors have been selected from the word×document co-occurrence matrix and have been transformed and combined to produce the sense vectors. The sense vectors of the ambiguous word contain the discrete words selected from its sense files. Each word in the sense vector has been assigned with a numeric vector selected from the word×document co-occurrence matrix. The dimension of this word vector is equal to the number of the senses of the ambiguous word. The word vectors of each sense file have been combined to produce the sense vector of the word. The combination methods such as concatenation, average, fractional and exponential decay combination strategies have used to create the sense vectors from the word vectors (Iacobacci, et al., 2016). The details of these integration strategies are as follows.

a) Concatenation method: The concatenation combination method aggregates the word vectors surrounding the ambiguous word in an instance into a larger vector. The words surrounding the target word have collected using the fixed window size *W*. The word vectors for *W* words on both sides of the target word are added to produce the sense vector for the instance. Equation (6.1) calculates the sense vector I_i for an instance of an ambiguous word by adding all the word vectors w_k in a window size *W*. The word vectors for target and surrounding words have been selected from the co-occurrence matrix.

$$I_i = \sum_{\substack{j=i-W \\ j \neq i}}^{i+W} w_{kj} \quad \ldots (6.1)$$

b) Average method: The average combination method computes the centroid of the word vectors of all the surrounding words of the target word. The window size *W* around the target word returns the surrounding words. Equation (6.2) explains the formula for the average combination method in which the sense vector I_i for an instance of an ambiguous word has been calculated by dividing the summation of all the word vectors w_i in a window size W by the total number of words in the window size that is *2W*.

$$I_i = \sum_{\substack{j=i-W \\ j \neq i}}^{i+W} \frac{w_{kj}}{2W} \quad \ldots (6.2)$$

c) Fractional decay: The fractional decay combination method combines the word vector of the surrounding words of the target ambiguous word by assigning weights to the word vectors. The weights assigned to the surrounding words are inversely proportional to their distance from the target word. Hence, the nearest surrounding words have assigned higher weights while the words that are far away have assigned less weight. Equation (6.3) calculates the weight of sense vector I_i by computing the weights of the word vectors of the ambiguous words.

$$I_i = \sum_{\substack{j=i-W \\ j \neq i}}^{i+W} w_{kj} \frac{W-|i-j|}{W} \quad \ldots (6.3)$$

d) Exponential decay: The exponential decay combination method also assigns the weights to the word vectors of the surrounding words of target words and then perform the addition to produce a resultant vector. The exponential decay combination method gives more importance to the close context. Therefore, the

weight parameter exponentially decreases with distance from the target word. In equation (6.4), the value of decay parameter α is 0.1.

$$I_i = \sum_{\substack{j=i-W \\ j \neq i}}^{i+W} w_{kj} \, (1-\alpha)^{|i-j|-1} \quad \ldots (6.4)$$

The ML classifiers have trained on the obtained sense vectors of the ambiguous Punjabi noun word along with their labels. The trained model has been used to predict the sense of the ambiguous word in the test instance.

(vii) **Prediction-based semantic word vectors:** The prediction-based semantic features are the numeric word vectors of the fixed dimension extracted by three different neural network models (Word2Vec, GloVe, FastText) from the unlabelled Punjabi corpus. The advantage of word embedding features is that they are extracted from the unlabelled dataset without any manual intervention. These word embedding features do not need any transformation. The word embedding features of the vocabulary words have a fixed size or dimension. Therefore, they need a few computational resources as compared to the count-based word×word vectors. Primarily these features have been used by the deep learning classifiers. Nevertheless, in this work, word embedding features have also experimented on the ML classifiers. The training dataset for the ML classifiers has been derived from the word embedding features. For the words of an instance of the ambiguous word, the word embedding vectors have been selected. The instance vector has been created by applying the concatenation, average, fractional and exponential decay combination methods on the word vectors.

The numeric training datasets of the 150 ambiguous Punjabi noun words have been used to train the machine learning and deep learning classifiers. The subtle details of the training of the classifiers are as follows.

6.1.2 Training of Classifiers

The supervised ML classifiers have been individually trained on the transformed sense tagged dataset. The ML classifiers have trained on the numeric trained dataset yielded by the topical, local, syntactic and semantic features. Whereas the deep learning classifiers have been trained on the word index vector and prediction-based word embedding features. The training of the models has been performed on the computer machine with the specifications mentioned in Table 6.1. Further, the implementation toolkit, training methodology and the parameters optimization of the machine learning and deep learning classifiers are as follows.

Table 6.1: Hardware configuration of the computer machine

Component	Value
Processor	Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz, 2801 MHz, 4 Core(s), 8 Logical Processor(s)
RAM	16GB
GPU	Nvidia GeForce GTX 1050
Operating system	64-bit Windows OS, x64-based processor

6.1.2.1 Training of the Machine Leanings Classifiers

The nine ML classifiers, namely Decision List, Decision Tree, Random Forest, Naïve Bayes, Sequential Minimal Optimization, k-Nearest Neighbour, AdaBoost, Stacking and StackingC have been selected from the machine learning toolkit WEKA (Waikato Environment for Knowledge Analysis) version 3.8.2 (Witten, et al., 2011). The WEKA toolkit is an open-source platform that has implemented machine learning algorithms in the JAVA programming language. The WEKA toolkit includes both the GUI and text interface to access the classifiers. In this work, the command line text scripts have been applied for the training of the different ML classifiers.

In the first step, the training dataset of an ambiguous word has collected in the comma-separated variable (csv) file format. The (csv) file has been converted into attribute relation file format (arff) file format suitable for classifier in the WEKA toolkit. The (arff) file format has been used in the training of the WEKA classifiers. In the second step, the individual ML classifiers have been trained on 26 training datasets (one BoW, one co-occurrence, two collocation combinations, three bigram combinations, five PoS combinations, one count-based word×document, word index vector and twelve word embedding features) of the ambiguous word.

Finally, the trained models of the individual ambiguous words have been saved. These saved models have been used to predict the sense of the test instance. These steps have been repeated for all the ML classifiers to produce the trained models for the 150 ambiguous words.

The training of ML classifiers has been validated by the application of the 10-fold cross-validation a statistical method of evaluation. The default number of iterations in the cross-validation method to get statistically meaningful results is 10. In the 10-fold cross-validation technique the training data is segregated into ten equal folds, and one fold, in turn, is used for testing, and the remaining nine folds are used for training ten times. The results are the average of ten-fold results.

The default parameters set of each WEKA classifier have been assigned during the training of the classifiers. (Amancio, et al., 2014) have experimented with the different WEKA classifiers, and based on the empirically obtained results from their experiment, authors have stated that default parameters yield the best results for the classification task.

6.1.2.2 Training of Deep Learning Classifiers

Two deep learning approaches, namely the Multilayer Layer Perceptron (MLP), a multilayer feedforward neural network and Long Short Term Memory (LSTM), have been applied for Punjabi WSD. The MLP and LSTM classifiers have been trained using the word embedding features of the ambiguous Punjabi words. The MLP and LSTM classifiers have been defined and built using the Keras Application Program Interface (API) written in Python language (Chollet, 2015).

Table 6.2: Libraries explored for the training of the deep learning classifier

Library	**Version**	**Description**
CUDA Toolkit	9.0.176	This toolkit consists of CUDA-C and CUDA-C++ compilers, IDEs, debuggers, profilers, utility tools and scientific, utility libraries. This library is required for the TensorFlow library to enable it to perform the CUDA GPU processing.
CuDNN	7.6.5.32	The CuDNN libraries are an update to CUDA for Deep Neural Nets and used by TensorFlow to accelerate deep learning on NVIDIA GPUs.
TensorFlow	1.12.0	TensorFlow is an end to end, Python friendly and open-source platform for the development of deep neural ntworks applications. It is developed by Google. Google has released it on 2015 under Apache Licence 2.0. TensorFlow accepts data in the form of multi-dimensional arrays of higher dimensions called tensors. TensorFlow works on the basis of data flow graphs that have nodes and edges. Thus TensorFlow code is easy to executed in a distributed manner.
Keras	2.1.6-tf	Keras library is written in Python language and it runs on the top of the machine learing platform TensorFlow. It is a simple, flexible and powerful deep learning API and enables the researchers to implement the deep learning neural networks quickly.

The Keras library has been used because of its modular, easy to extend and user-friendly nature. In the backend of the Keras API, the open-source machine learning platform TensorFlow has been applied. TensorFlow and Keras libraries are called the building blocks of deep learning classifiers. To harness the capabilities of the NVIDIA graphic card in the training of the MLP and LSTM classifiers, the CUDA toolkit and CuDNN libraries have been installed and

configured. The libraries, their versions and descriptions applied for the training of the MLP and LSTM classifiers have been given in Table 6.2. The MLP and LSTM classifiers have been designed using sequential Keras models. The configurations applied for the training of the MLP and LSTM classifiers are described as follows.

(i) The MLP classifier has been prepared using the stacked layers of the Keras API. The first layer is called the input layer, followed by two hidden layers, and finally, the output layer is included in MLP. The detailed configuration of the input, hidden, and output layers of the MLP classifier are as follows.

- **Input layer:** The first layer of the MLP classifier is the input layer which receives the input data. The input layer that is used for text related inputs is called the embedding layer in the Keras API. There are two different configurations of the input embedding layer of Keras API. In the first method, the input embedding layer has learned the word embedding features along with the training of the MLP classifier. In the second method, the pre-trained word embedding features have been loaded during the training of the deep learning classifiers. The use of pre-trained word embedding not only reduces the learning time for the classifier but also provides more pertinent weight vectors for the words. Therefore, in this work, the pre-trained word embedding features have been loaded in the input embedding layer during the training of the MLP and LSTM classifiers. Three different word embedding features the Word2Vec, GloVe and FastText extracted from the unlabelled Punjabi language text, were considered. The Word2Vec, GloVe and FastText word embedding features are the numeric word vectors with fixed dimension. The parameters of the input embedding layer are shown in Table 6.3. The output shape of the input embedding layer is a 3D tensor (batch_size, input_length, output_dim) which has been fed to the hidden layers.

Table 6.3: Parameters of the input layer

Parameter	**Value**	**Description**
Input dimension	Unique words in sense tagged dataset	Total size of the vocabulary
Output dimension	100/200/300	The size of the vector space for the words
Input length	50	The length of input sequences
Weights	Word2Vec/GloVe/ FastText word vectors	The pre-trained word embedding using Word2Vec/GloVe/FastText NN models

- **Hidden layers:** Two hidden layers of the MLP classifier have been stacked one over another and configured in this work. The output of the input embedding layer is a 3D

tensor that must be flattened into a single vector so that values can be directly passed to the hidden layer. Therefore, a flatten layer defined in Keras API has been inserted in between the input embedding layer and the immediate hidden layer. The flatten layer transforms a 3D tensor into a vector that can be fed to a fully connected hidden layer. Two hidden layers have dense connections between them. In dense connections, each neuron of one hidden layer is connected with all neurons of the other hidden layer. The dimensions of both the hidden layers are 200. The activation function Rectified Linear Unit (ReLU) has been applied at both the hidden layers. The 200-dimensional output of the second hidden layer is connected to the output layer of the MLP classifier.

- **Output layer:** The size of the output layer is equal to the number of senses of the ambiguous word. The output layer uses the softmax activation function to finally identify the sense of the ambiguous word.

(ii) The LSTM classifier has been trained on the predictive word vectors. Figure 6.2 explains the basic layered setup of the LSTM classifier using the Keras API sequential model. The configuration of the LSTM model applied for the Punjabi WSD is as follows.

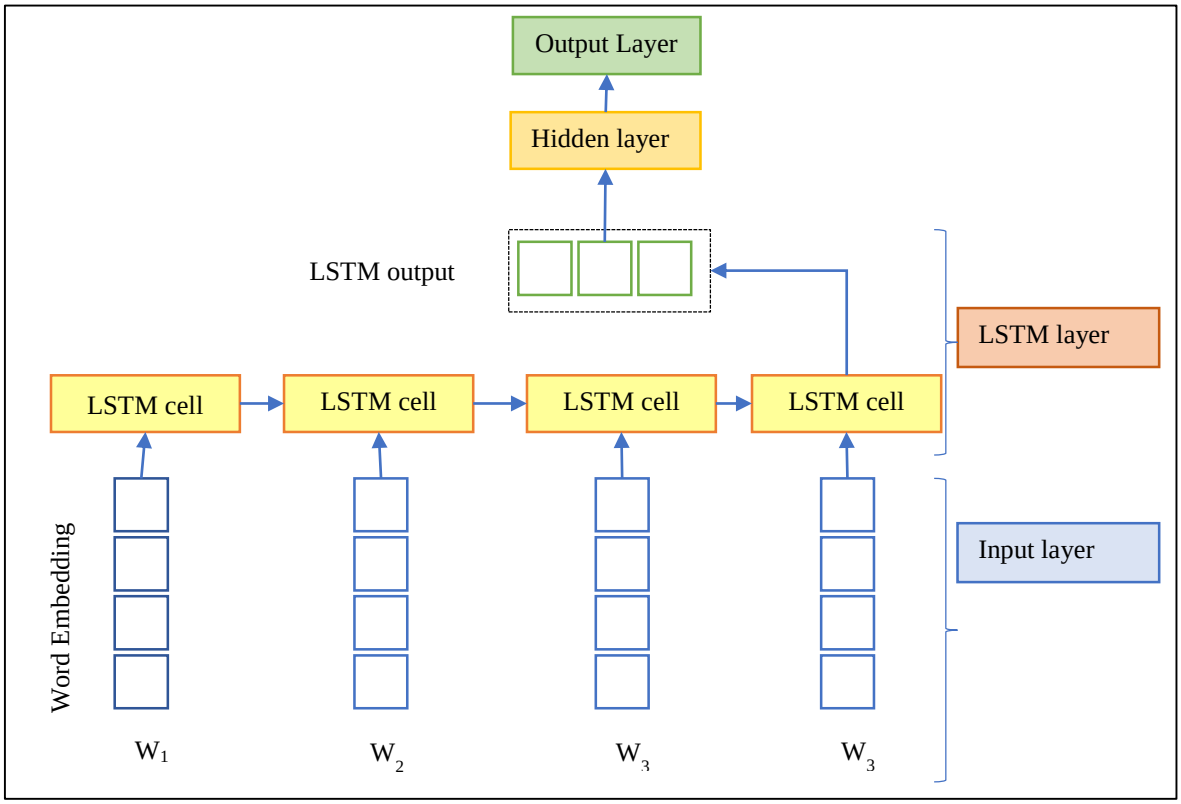


Figure 6.2: The setup of the LSTM model

- **Input layer:** The input layer of the LSTM model is the first layer of the model. The input layer parameters for the LSTM model are the same as in the MLP model. The input layer of the LSTM model explores the word embedding features Word2Vec,

GloVe and FastText. The parameter input dimension is the vocabulary of the ambiguous word that is the collection of unique words selected from its sense tagged dataset. The output dimension of the words is the pre-trained word embedding dimension. The real number value of the word vector forms its weight. The input length of the embedding layer is the length of the sequence of the input. The input parameters are the same as defined in the Keras input embedding layer of MLP classifier. The trainable attribute of the embedding input layer of Keras is false because the pre-trained embedding matric weights have been used in this model.

- **LSTM Layer:** The LSTM layer consists of LSTM gates, and it is stacked over the input layer. In this work 128, gated memory units are defined in the LSTM Layer. Before passing the output to the hidden layer, the output has been passed through the dropout layer. The dropout layer helps in reducing the overfitting problem of the model.
- **Hidden layers:** The output from the LSTM layer has been given to the dense hidden layers that are stacked one after another. The output of the LSTM layer is a 3D tensor that must be flattened into a single vector so that values can be directly passed to the hidden layer. The dimensions of both the hidden layers are 200. The ReLU activation function has been applied as an activation function at both the hidden layers. The output of the second hidden layer is fed to the output layer of the LSTM model.
- **Output layer:** The dimension of the output layer is equal to the number of senses of the ambiguous word. The output layer uses the softmax activation function to identify the sense of the ambiguous word.

After defining the configuration of the deep learning models, these models have been compiled and trained. The compilation parameters that have been configured for training and testing of MLP and LSTM classifiers are given in Table 6.4. Initial weights between different layers are uniform for both models. The Adam optimization is applied to the models to update the weights among the different layers. Cross entropy, a multiclass logarithm function has been applied as a loss function by both models. The number of training instances of an ambiguous word applied to the input layer before updating weights is ten, and the training of these models is repeated 50 times (epoch). The default learning rate values of the keras library optimizers were applied for both the MLP and LSTM models.

To validate the individual classification models, split validation has been applied. The split validator partitioned the same sense tagged dataset into two subsets, in the ratio of 80:20. Both

the models are trained on 80% of the sense tagged dataset and then tested on the remaining 20% dataset.

Table 6.4: Parameters used by deep learning models

Parameters	MLP	LSTM
Model initialization		
Hidden layers	2	2
Hidden units	100 and 200	Gated memory units 128
Layer Types	Dense (fully connected)	
Weight initialization	Uniform	Uniform
Activation function	ReLU, softmax	Sigmoid, tanh, softmax
Model Compilation		
Optimizer	Adam (adaptive moment estimation)	
Loss Function	Categorical cross entropy	
Metrics	Accuracy	
Model Training		
Epoch	50	
Batch size	10	

The MLP and the LSTM neural network models have been prepared for the individual ambiguous word of the Punjabi language. Three different word embedding features of the individual ambiguous word have been applied for the training of the MLP and LSTM classifiers. The trained models of the supervised classifiers have been saved for the individual Punjabi ambiguous noun word. These models can be used to predict the sense of the test instance.

6.1.3 Predicting the Sense of the Ambiguous Word

The trained models prepared in the first step of the classification have been used to predict the sense of the ambiguous word in a test instance. The detailed process of identification of sense of an ambiguous word using the ML trained models has been elaborated in Figure 6.3 using a flowchart.

(i) The input to the trained models is the Punjabi instance. The size of the instance can be a paragraph or a single instance.

(ii) The instance is tokenized, pre-processed, normalized, and the unique tokens have been generated.

(iii) The individual word tokens have been checked in the dataset of the 150 Punjabi ambiguous noun words to check if the tokens exist in the labelled dataset of the Punjabi language.

(iv) If a word does not exist in the dataset of 150 ambiguous Punjabi noun words, a message is displayed that no ambiguous word is found in the sentence. Otherwise, the ambiguous tokens that exist in dataset have been processed.

(v) The identified ambiguous words have been inserted into a queue. The ambiguous words from the queue have been processed independently.

(vi) The context words of the identified ambiguous word act as the features of the test instance.

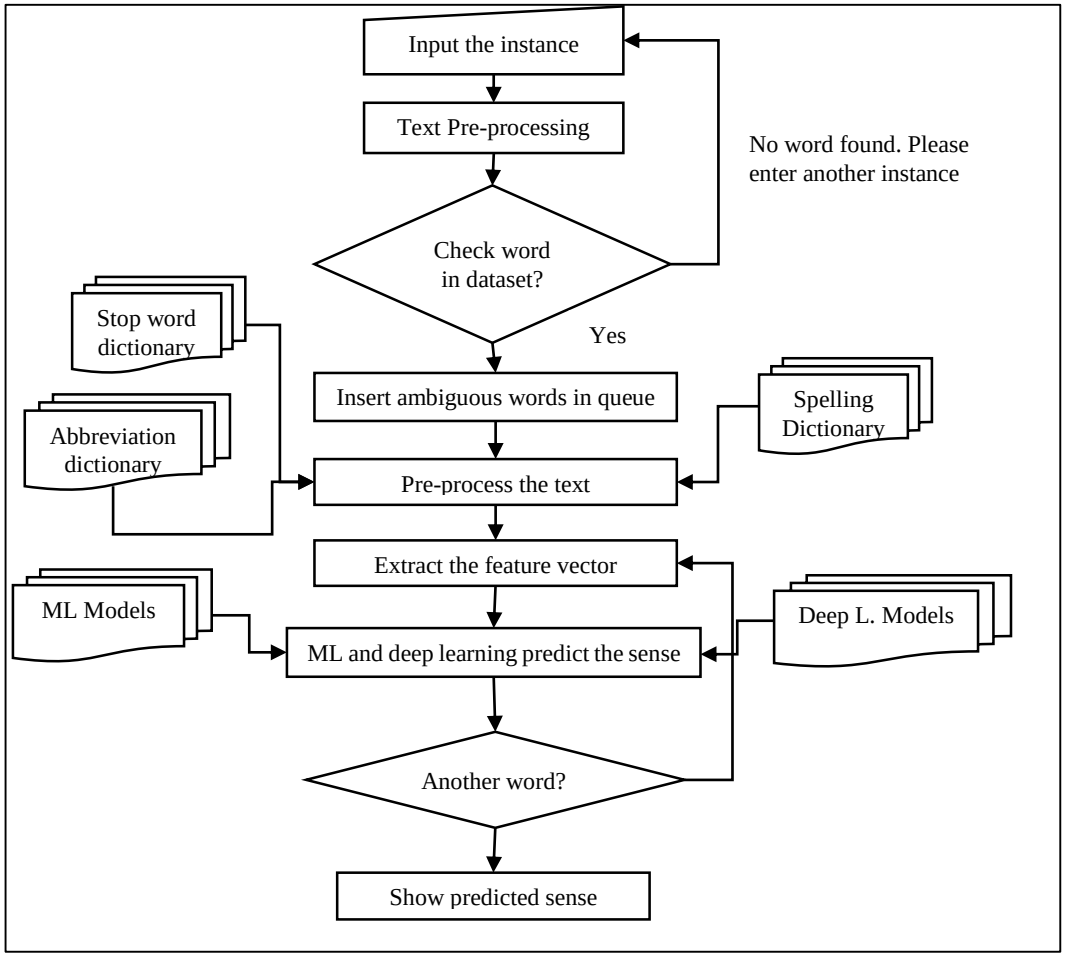


Figure 6.3: Flowchart of predicting sense of the ambiguous words using trained classifiers

(vii) The test word vector has been compared with the training word vectors of the ambiguous word and has been transformed into a numeric vector by the application of string to vector transformations. The BoW string to vector transformation converts the test string vector into a numeric weighted vector. Similarly, the real number co-occurrence vector has been derived from the co-occurrence vector. The binary test vector has been prepared by comparing it with the feature vectors of the collocation, bigram and PoS combinations. However, the test vector using the semantic feature has been derived by selecting the individual test word vector from the word×documnet, and the word embedding representations. The test word vectors of the test instances are

combined by one of the four combination strategies, namely concatenation, average, fractional and exponential.

(viii) The prepared test word vector has been supplied to topical, local, syntactic and semantic trained models of ML and deep learning classifiers that predict the sense of the ambiguous word.

(ix) This process has been repeated until senses for all ambiguous words in the sentence have been predicted by the classifiers.

The classification task of the WSD yields the sense of the ambiguous Punjabi word in the context. The Punjabi WSD architecture has been implemented as a web application. The details of the integration of the Punjabi WSD is given in the following section.

6.2 Hosting of the Punjabi WSD on Web

The different components of the Punjabi WSD architecture have been integrated with the web. This web-based system has been implemented in the apache tomcat 9.0 server. The web interface has used the JAVA, JSP for server-side programming and the ML classifiers of the WEKA. The Keras API and Python scripts have been used to implement the MLP and LSTM models.

6.2.1 Data Structures Used in the Implementation of the Punjabi WSD

The different phases of the Punjabi WSD architecture have stored the data in the different data structures that are discussed as follows.


```
{
"word": "ਯੋਗ",
"Sense": [ "ਕੁਸ਼ਲ ਕਾਬਲ ", "ਵਰਨਣ ਲਈ ਠੀਕ","ਵਰਜਿਸ ਅਤੇ ਧਿਆਨ"],
"Example": [ "ਤਕਨੀਕੀ ਕੰਮ ਲਈ ਯੋਗ ਬੰਦਾ ਲਭੋ" , "ਯੋਗ ਠੀਕ ਹੋਣ ਦੀ ਅਵਸਥਾ ਜਾਂ ਭਾਵ", "ਯੋਗ ਨਾਲ ਸ਼ਰੀਰ ਅਰੋਗ ਰਹਿੰਦਾ ਹੈ"]
} ,
{
"word": "ਜੂਨ",
"Sense": [ "ਮਹੀਨਾ ", "ਜਾਨ"],
"Example": [ "ਜੂਨ ਦਾ ਮਹੀਨੇ ਵਿਚ ਗਰਮੀ ਹੁੰਦੀ ਹੈ " , "ਕਿਸਾਨ ਦੀ ਤਾਂ ਜੂਨ ਹੀ ਖਰਾਬ ਹੈ "]
} ,
{
"word": "ਜੱਗਾ",
"Sense": [ "ਇਕ ਧਾਰਮਿਕ ਇਕੱਠ ", "ਭਾਂਡਾ","ਲੋਕ", "ਅਖਬਾਰ"],
"Example": [ "ਪਿੰਡਾਂ ਵਿੱਚ ਕਈ ਵਾਰ ਜੱਗਾ ਕੀਤਾ ਜਾਂਦਾ ਹੈ " , "ਜੱਗਾ ਲੈਂਟੇ ਵਰਗਾ ਇਕ ਭਾਂਡਾ", "ਜੱਗਾ ਜਿਉਦਿਆਂ ਦੇ ਮੇਲੇ ਹਨ" , "ਜੱਗਾ ਬਾਣੀ ਅਖਬਾਰ ਦਾ ਨਾਮ ਹੈ"]
} ,
{
"word": "ਜੰਗਾ",
"Sense": [ "ਲੜਾਈ ", "ਜਰ ਲੱਗਣਾ ","ਨਾਮ"],
"Example": [ "ਜੰਗ ਅਠਾਰਹਾ ਦਿਨਾਂ ਤੱਕ ਚਲੀ" , "ਲੋਹੇ ਨੂੰ ਜੰਗ ਆਸਾਨੀ ਨਾਲ ਲੱਗ ਜਾਂਦੀ ਹੈ", "ਦਿੱਲੀ ਦੇ ਉਪ ਰਾਜਪਾਲ ਨਜੀਬ ਜੰਗ "]
} ,
{
"word": "ਜਾਂਚ",
"Sense": [ "ਪਤਾ ਕਰਨਾ ", "ਨਿਰੀਖਣ ","ਛਾਣ ਬੀਨ"],
"Example": [ "ਟੀਬੀ ਦੀ ਜਾਂਚ ਅਤੇ ਇਲਾਜ" , "ਪੁਲਿਸ ਦੀ ਜਾਂਚ-ਪੜਤਾਲ ਵਿੱਚ ਮਦਦ ਕਰ", "ਇਸ ਮਾਮਲੇ ਦੀ ਜਾਂਚ ਸੰਭਾਲਣ ਦੀ ਪ੍ਰਕਿਰਿਆ ਵਿੱਚ ਹੈ "]
} ,
{
"word": "ਜਾਲ",
"Sense": [ "ਇਕ ਤਰ੍ਹਾਂ ਦੀ ਤੋਪ", "ਜਾਲ ਇਕ ਵਿਚ ਬੁਣੀ ਹੋਈ","ਚਾਲ"],
"Example": [ "ਦੁਸ਼ਮਣਾਂ ਨੇ ਜਾਲ ਨਾਲ ਕਿਲੇ ਨੂੰ ਢੇਰੀ ਕਰ ਦਿਤਾ" , "ਇਸ ਤਰ੍ਹਾਂ ਵਿਛੇਗਾ ਸੜਕਾਂ ਦਾ ਜਾਲ", "ਰੋਜ਼ਗਾਰ ਦੇ ਜਾਲ ਵਿੱਚ ਫਸਣ ਤੋਂ ਬਚੋ "]
} ,
```


Figure 6.4.: JSON file structure used to store the Punjabi ambiguous words and their coarse senses

(i) The JavaScript Object Notation (JSON) structure is a name/ pair structure that has stored the 150 ambiguous Punjabi noun words, their senses and their description examples. Figure 6.4 shows the data stored in the JSON file. To add the details of a new ambiguous word to the sense tagged dataset, this file has to be updated. The JSON file has been searched by the Punjabi WSD system to determine if the tokens of the test instance consist of any ambiguous word.

(ii) The preprocessing dictionaries such as spelling, stop words and abbreviation dictionaries are stored in plain text files. The spelling file consists of a word and its multiple spelling words in a single row separated by tab space. The stop word list is a file of stop words having a single word per row. The pair of abbreviations and its detail is saved in the tab-separated file.

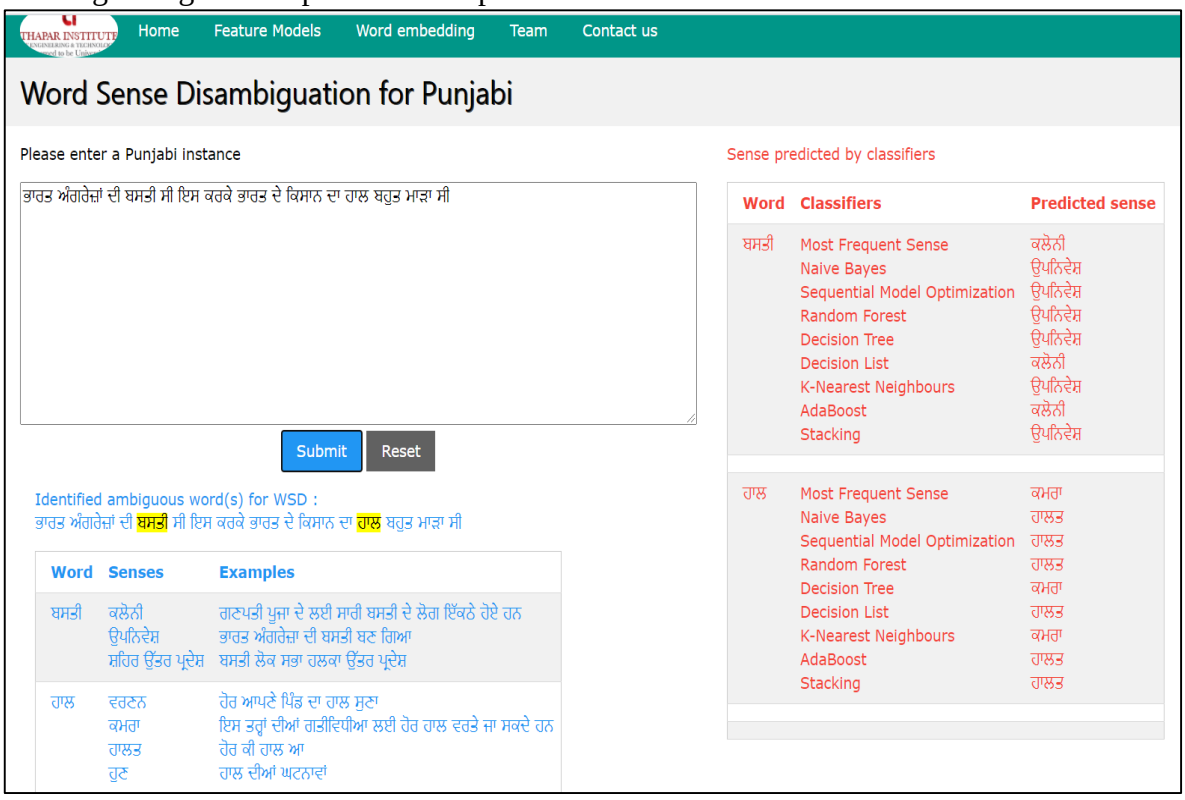


Figure 6.5: Sense predicted by the Punjabi WSD

(iii) The topical, local semantic and syntactic features files for the individual Punjabi word have prepared and stored in (csv) file format. However, to make these files compatible with the WEKA toolkit, these files have been converted into (arff) file format to train the ML classifiers in the WEKA toolkit. The test instance is converted and stored into the (arff) file format.

(iv) The individual ML classifiers trained on the different features of the ambiguous Punjabi word. The trained models have been saved into a model file. The saved ML model file is

used to predicts the sense of the ambiguous word in the test instance. It uses the arff file format of the test instance.

(v) The word embedding features have been stored as a word value paired dictionary. This dictionary can be accessed from word embedding. The MLP and LSTM classifiers have built using these word embedding features. The embedding feature files have been used to determine the most similar words, analogies and the similarity operation.

6.2.2 The Punjabi WSD Interface

The Punjabi WSD interface can be accessed using www.punjabiwsd.in. The description and functioning of the web-based Punjabi WSD have been discussed with the help of the following figures.

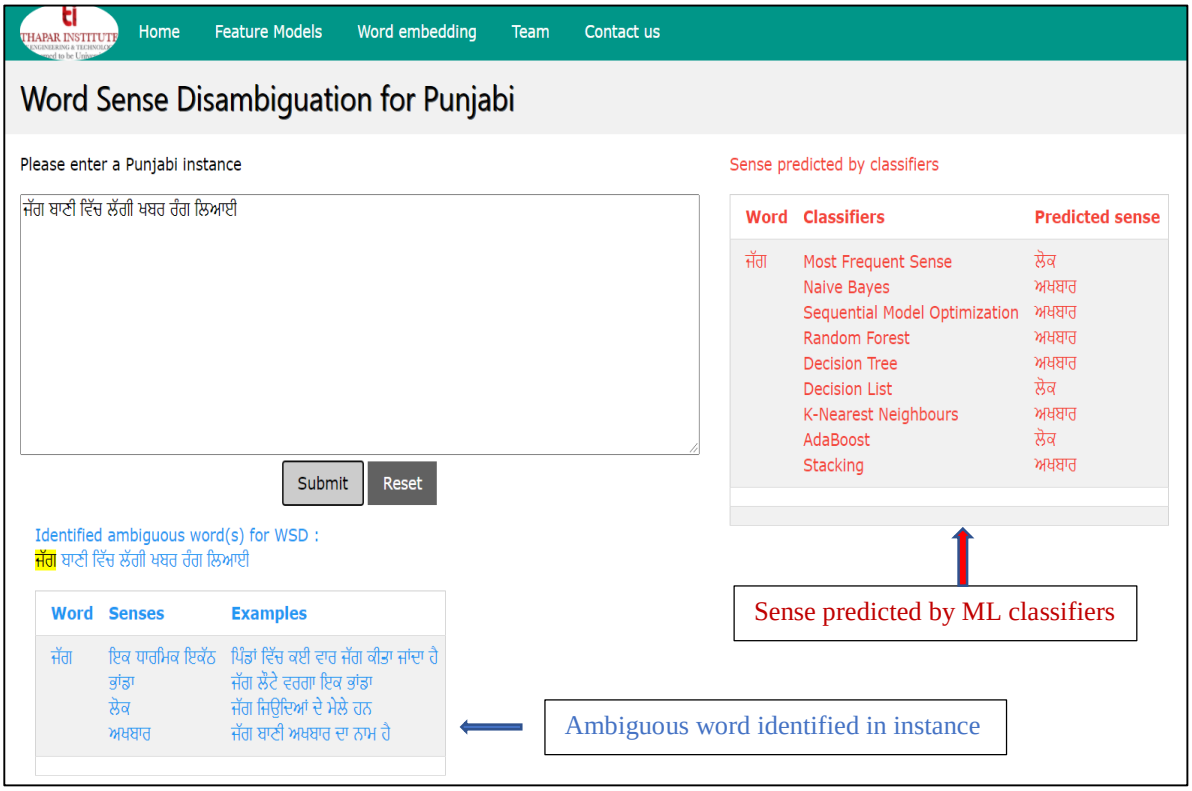


Figure 6.6: Sense predicted by the Punjabi WSD of multiple words

(i) The Home page of the Punjabi WSD system is shown in Figure 6.5. On this page, the user has an option to insert the Punjabi text instance in the text area. The Punjabi text can be a sentence or a paragraph. On clicking the submit button, the Punjabi WSD system process the Punjabi text of text area in the background and returns the outputs in the form of two tables. The first table (blue text) highlights the ambiguous word (ਜੱਗ) present in the Punjabi instance inputted by the user. It also shows the senses and the example sentences of the highlighted ambiguous word. The second table (red text) shows the sense of the identified

ambiguous word predicted by all the ML classifiers using the BoW feature sets. The Punjabi WSD system preprocesses the Punjabi text in the background, searches the Punjabi tokens in the JSON file, prepares the BoW features of the identified ambiguous word and gives it to trained model files of different ML classifiers. The trained ML classifiers predict and return the sense of the ambiguous Punjabi word. If the token words of the Punjabi test instance do not exist in the Punjab ambiguous word database, then the message 'word not found' is printed. The reset button in Figure 6.5 clears the text area.

(ii) The Punjabi WSD system can handle more than one ambiguous word present in the Punjabi text entered by the user. In case the Punjabi text contains more than one ambiguous word, the Punjabi WSD system stores the identified multiple Punjabi ambiguous words in a queue as per their sequence in the sentence. The ambiguous Punjabi words have preprocessed individually and the ML trained models disambiguate the ambiguous words one by one. The output of the disambiguation of multiple ambiguous words present in the single Punjabi sentence is shown in Figure 6.6. The Punjabi WSD system in Figure 6.6 has identified two Punjabi ambiguous words ਬਸਤੀ (TT: bastī) and ਹਾਲ (TT: hāl) in the input Punjabi sentence.

The senses of these two words have been predicted independently and are shown in red text tables in Figure 6.6.

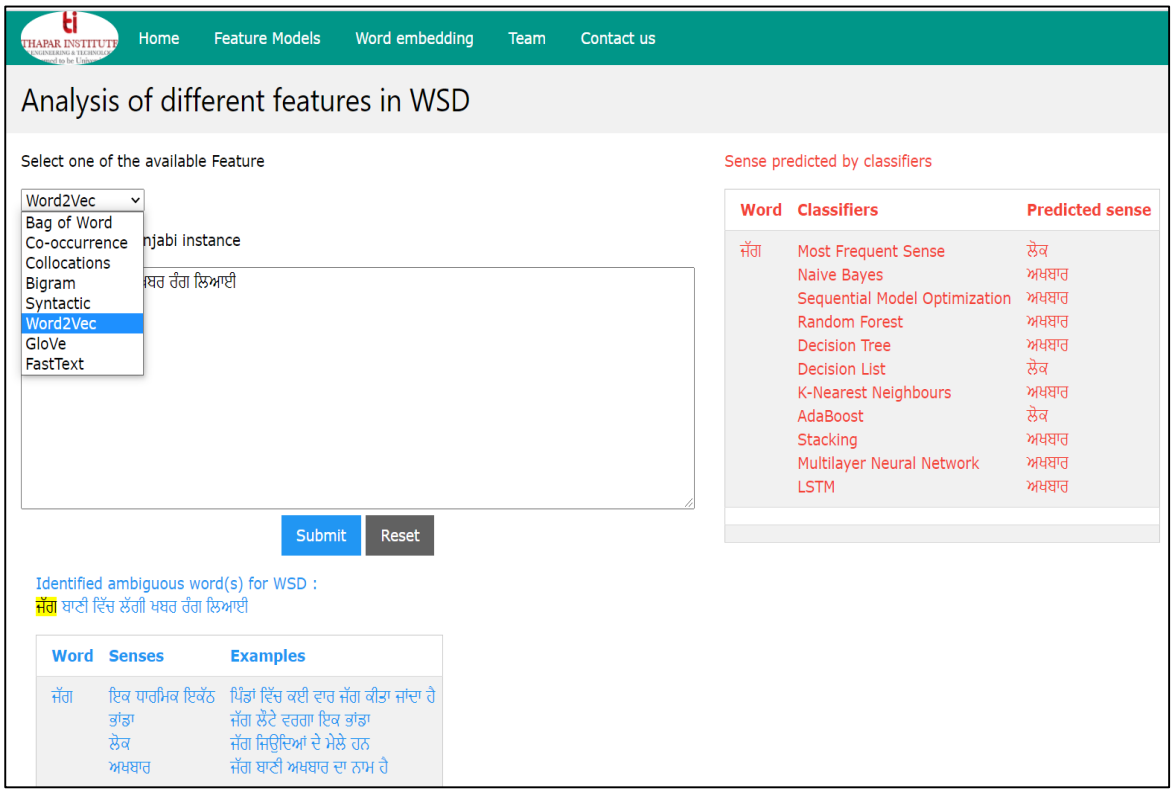


Figure 6.7: Feature specific Punjabi WSD

(iii) The Punjabi WSD system, by default, has achieved the sense classification of ambiguous words using the ML classifiers based on the BoW features. However, the feature models tab at the home screen gives the user an option to select the different features, namely topical, local and semantic features, as shown in Figure 6.7. The user can opt for any one of the co-occurrence, collocation, bigram, semantic (TF-IDF), Word2Vec, GloVe, FastText features to perform the Punjabi WSD. After selecting the suitable feature, the user can input the Punjabi instance in the text area. The Punjabi WSD system will perform the classification of the sense of the ambiguous word based on the selected feature. The MLP and LSTM classifiers have also trained on the word embedding features. When one of the word embedding features such as the Word2Vec, GloVe or FastText have selected, both the MLP and LSTM models alongwith other ML classifiers predict the sense of the

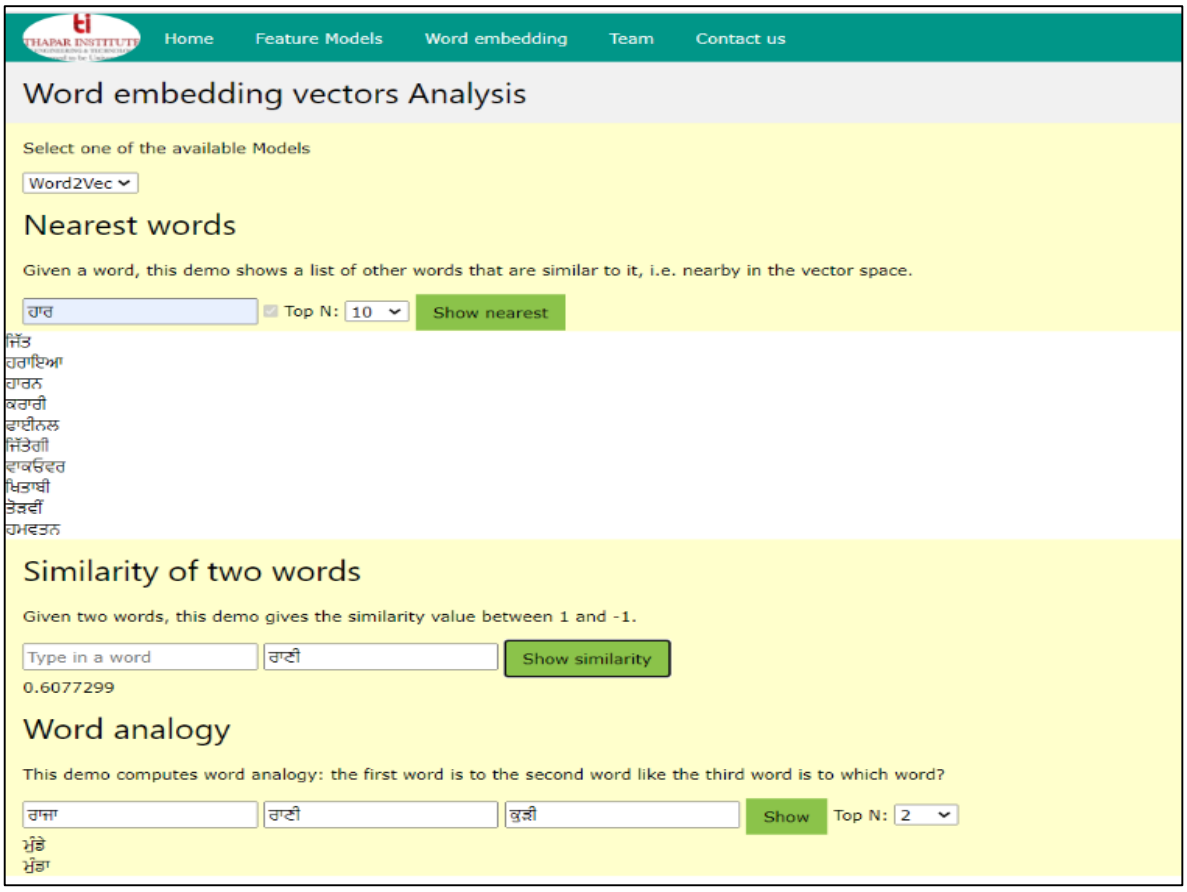


Figure 6.8: Analysis of the Word embedding features

ambiguous Punjabi word. Figure 6.7 illustrates the output produced by the ML, MLP and LSTM classifiers of the Punjabi WSD system using Word2Vec feature set.

(iv) In addition to the Punjabi WSD system, the evaluation of the word embedding features can also be performed using this web interface. Three tasks nearest word, the similarity of two words and word analogy can be performed on the different word embedding features of the

Punjabi language words. Figure 6.8 gives details of the analysis of the word embedding features. The user can opt for one of the Punjabi Word2Vec, GloVe and FastText word vectors derived from the Punjabi language unlabeled corpus. The nearest words task returns the list of 10 nearest words to the input Punjabi word based on the Word2Vec word vector feature. The similarity word task returns the cosine similarity between the two Punjabi words entered by the user. The analogy task is like the first word is to the second word, and the third word is to which word. It returns the two most analogues words for the fourth word. These word vectors are also available for the users to further explore them in other NLP tasks of the Punjabi language.

The availability of the proposed research work as web appliaciotn will help the users to explore the Punjabi WSD system. The users can integrate this Punjabi WSD application into their NLP applications.

Chapter Summary

The classification of the ambiguous words has been performed by the ML and deep learning classifiers. The classifiers have been trained on the selected features such as topical, local, syntactic and semantic features selected from the context of the Punjabi sense tagged. However, the features selected from the Punjabi text consist of Punjabi words. To train the ML classifiers the string features have been transformed into numeric word vectors. Individual algorithms have been developed to transform the topic, local and syntactic features. The numeric features have been used by the ML classifiers for training.

Nevertheless, the semantic count-based and predictions based word representations have numeric co-occurrence matrices of the Punjabi language words. To apply these word vectors for the training of the ML classifiers, these word vectors have been combined to generate sense vectors using concatenation, average, fractional decay and exponential decay combination methods. These sense vectors have been used by the ML classifiers to build the respective models. The prediction-based word vectors have been directly explored by MLP and LSTM classifiers to prepare the trained models.

The ML classifiers of the WEKA toolkit have been used in this work. The MLP and LSTM neural network models have been designed and developed using the Keras library. The 10-fold cross-validation training method has been applied to perform the training of the ML classifiers. While the MLP and LSTM classifiers have trained using the 80:20 split strategy. The data structure used to store the ambiguous words and the Punjabi dictionaries used in the Punjabi WSD has been presented in this chapter. Finally, the trained models have been integrated with the web interface of the Punjabi WSD.

Chapter 7 : Results and Discussions

This chapter presents the results and discussion on the work carried out in this thesis. The details of the proposed architecture of the Punjabi language Word Sense Disambiguation (WSD) have been described in the earlier chapters. The sense tagged dataset has been created and processed to select the topical, local and semantic features for the 150 Punjabi ambiguous noun words. The Punjabi language raw corpus has been used to create the word embedding features. The machine learning and deep learning classifiers have implemented the Punjabi WSD using these features for the individual Punjabi noun words. The evaluation of the Punjabi WSD based on the machine learning and deep learning classifiers has been discussed in this chapter.

The performance of the WSD system for the Punjabi language has been estimated by calculating the accuracy metric for the individual ambiguous word. This chapter shows the estimated average accuracies of the nine machine learning and two deep learning classification techniques in conjunction with an empirical analysis for statistical and prediction-based feature extraction methods. The accuracies produced by machine learning and deep learning classifiers have been compared with the baseline Most Frequent Sense (MFS) accuracies of the individual ambiguous word. The detailed performance analysis of the machine learning and deep learning classifiers has been presented in subsequent sections.

7.1 Evaluation of WSD

The proposed automatic Punjabi WSD system has been evaluated as a standalone application. The Machine Learning (ML) and deep learning classifiers have trained on the features selected from the Punjabi language corpus. The metrics explored to determine the performance of the ML and deep learning classifiers of Punjabi WSD are as follows.

7.1.1 Evaluation Metric

The evaluation measure accuracy has been used to determine the performance of the Punjabi WSD system. The accuracy of the WSD system depends on the number of test instances of the ambiguous words correctly determined by it. The accuracy is also referred to as recall in the WSD research work (Navigli, 2009). The accuracy is estimated using equation (7.1). In equation (7.1), the accuracy of the WSD system is the number of correctly identified instances I with the right sense of the target word w having n senses to the total m number of instances present in the dataset of the target word.

$$Acc = \frac{|I_i^j; j \in 1,2,\ldots,m, i \in 1,2,\ldots,n|}{m} \quad \ldots (7.1)$$

The accuracy depends on the coverage of the WSD system. The coverage of the WSD system is defined as the ratio of the number of senses predicted by the system of the instances of the test dataset to the total number of instances in the test dataset and is given in equation (7.2)

$$Cov = \frac{\text{\#senses predicted for instances}}{\text{\#test data instances}} \quad \ldots (7.2)$$

The Punjabi WSD system has returned at least one sense for the target ambiguous word. Thus the coverage of the Punjabi WSD system is 100%. When the coverage is 100%, then the precision and the accuracy of the Punjabi WSD system become equal. The precision metric is the ratio of the correct senses assigned to the target word by the WSD system to the number of senses predicted by the system. The formula of precision of the WSD system is given in equation (7.3). It is estimated for an individual word. This Punjabi WSD has the same accuracy and precision as it assigns a sense to each target word of the individual instance of the dataset.

$$Pre = \frac{\text{\#correct senses predicted}}{\text{\#total senses predicted}} \quad \ldots (7.3)$$

The parameter F1 measure is the weighted harmonic mean of precision and recall (accuracy) as given in equation (7.4). It becomes one when the precision is equal to recall and thus cannot be used to validate results.

$$F1 = \frac{2 \times \text{Acc} \times \text{Pre}}{Acc + Pre} \quad \ldots (7.4)$$

Therefore, in this research work, the accuracy metric has been used to evaluate the performance of the Punjabi WSD. The performance of the Punjabi WSD system has been compared with a baseline. The baseline performance is the lower bound performance for the WSD system. The most frequent sense baseline method has been employed in this thesis.

7.1.2 Baseline Most Frequent Sense (MFS)

The baseline MFS assigns the most frequent sense in the training dataset of an ambiguous target word. The most frequent sense of an individual target ambiguous word is the sense having the largest number of instances among the different senses of target ambiguous word. It has been observed by the researchers that for WSD systems, it is difficult to beat the MFS baseline (Navigli, 2009). Equation (7.5) defines the MFS accuracy of the ambiguous words *w* having *n* senses.

$$MFS(w) = \frac{\max(S_i^w)}{\sum_i^n S_i} \quad \ldots (7.5)$$

The baseline MFS of the 150 ambiguous Punjabi noun words has been depicted in Figure 7.1. The horizontal axis of the graph in Figure 7.1 represents the 150 Punjabi ambiguous words, and the vertical axis represents the baseline MFS accuracy.

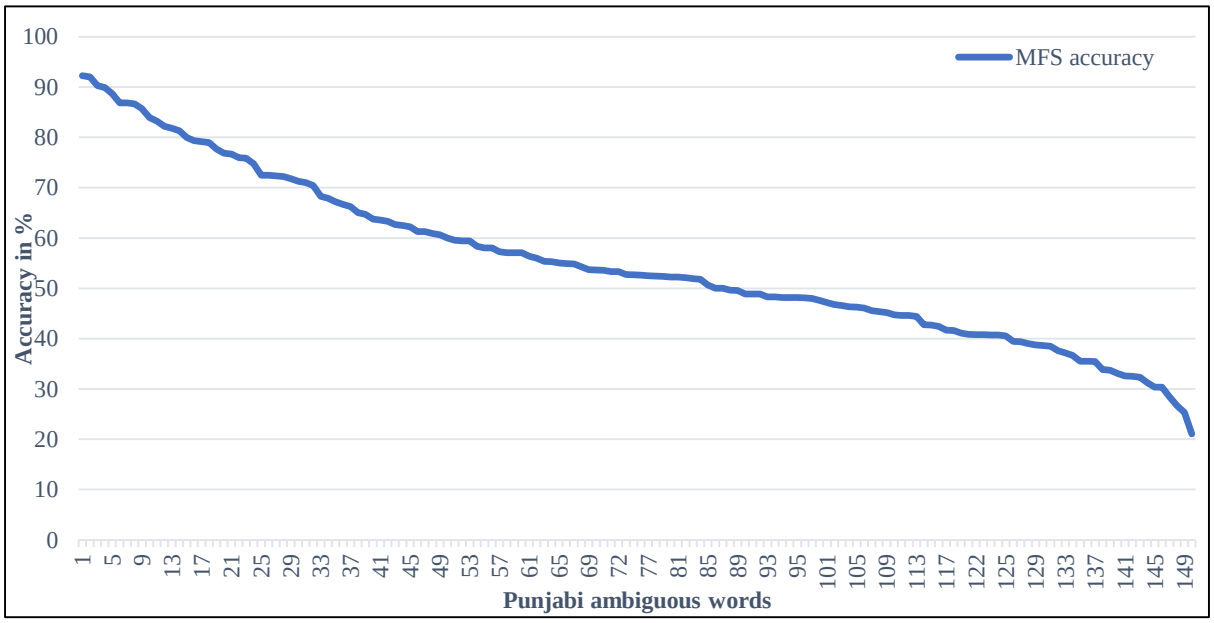


Figure 7.1:MFS accuracy of 150 ambiguous Punjabi noun words

In Figure 7.1, the 150 Punjabi ambiguous words have been arranged in the decreasing order of their MFS accuracy. The MFS sense of the individual Punjabi ambiguous noun word has been marked with the blue line, and it indicates the MFS instances of the Punjabi ambiguous words dominants the sense tagged dataset. On average, there exist 130 instances per ambiguous word, out of which 72 instances belong to the most frequent sense of the ambiguous Punjabi words. Thus the baseline accuracy that is the lower bound in this experiment is very high. The accuracy of the Punjabi WSD produced by ML and deep learning classifiers using different feature sets have been evaluated and compared with the MFS baseline.

7.2 Experimented Results of the ML Classifiers

The nine ML classifiers Decision List (DL), Decision Tree (DT), Random Forest (RF), Naïve Bayes (NB), Support Vector Machine (SVM) based on sequential minimal optimization, k Nearest Neighbour (k-NN), Adaboost (AB), Stacking (Stk), StackingC (StkC) have been trained individually on the topical, local, syntactic and semantic features selected form the sense tagged dataset of the 150 ambiguous Punjabi noun words. These nine supervised ML classifiers have also explored the word embedding features. The accuracy scores of the Punjabi WSD using the ML classifiers have been estimated for the individual Punjabi ambiguous word. The training of the ML classifiers has been validated using the stratified 10-fold cross-

validation approach. The statistically significant paired t-test with 10-fold cross-validation at the p-value of 5% in the WEKA software has been applied for the comparison of the accuracies of the ML classifiers using a different set of features with the baseline MFS accuracies. The detailed result analysis of the Punjabi WSD system using topical, local, syntactic and semantic features are discussed as follows.

7.2.1 Analysis of the Punjabi WSD using Topical Features

The average accuracies of the Punjabi WSD system using nine ML classifiers on the bag of words (BoW) and co-occurrence topical features of the 150 ambiguous Punjabi noun words is presented in Table 7.1.

Table 7.1: Average accuracies of the nine ML classifiers using the BoW and Co-occurrence features

Topical Features	**MLClassifiers**									
	DL	DT	RF	NB	SVM	k-NN	AB	Stk	StkC	MFS
BoW	79.17	80.07	85.12	86.75	84.56	69.80	81.73	87.98	**89.14**	52.36
Co-occurrence	79.06	80.49	84.96	81.94	82.24	70.36	81.53	84.39	85.17	

The experimental results indicate that the DL, RF, NB, SVM, AB, Stk and StkC classifiers have produced slightly better average accuracies using the BoW features of the Punjabi words. At the same time the DT and k-NN classifiers have yielded better average accuracies using the co-occurrence features. However, there exists no statistically significant difference among the accuracies of nine classifiers when BoW and co-occurrence features have been applied for the individual Punjabi noun word. The best performance has been produced by the ensemble classifiers that are StkC and Stk using BoW and co-occurrence feature sets for the Punjabi WSD.

It has been further observed that the rare classes have a high misclassification rate when base classifiers have been used. In comparison, the performance of the ensemble methods is better than the base classifiers in the case of rare sense class classification. The performance is independent of the features used by the ensemble WSD systems. The predictive performance of base and ensemble WSD on rare senses of ambiguous words is shown in Table 7.2. It shows the senses predicted by the best base model SVM and the ensemble model StkC when applied on BoW features of four ambiguous Punjabi words, each having four coarse senses. It has been observed that the ensemble classifier disambiguates rare senses of ambiguous words efficiently than the base classifier.

Table 7.2:Individual sense prediction by the SVM and StkC classifiers using BoW features of the Punjabi words

ਤਾਲ(tāl)			ਫਲ(phal)			ਮਾਪ(māp)			ਸੋਚ(sōc)		
Instances per sense			Instances per sense			Instances per sense			Instances per sense		
Actual	Predicted		Actual	Predicted		Actual	Predicted		Actual	Predicted	
	SVM	StkC		SVM	StkC		SVM	StkC		SVM	StkC
91	87	87	153	153	153	42	39	39	75	70	69
16	6	7	31	29	29	29	24	23	55	47	48
11	4	4	6	2	6	10	8	8	49	44	44
5	0	2	6	4	3	5	1	2	11	6	7
Acc.	78.8%	81.3%	Acc.	95.8%	97.4%	Acc.	83.7%	83.7%	Acc.	87.97%	88.5%
MFS	73.9%		MFS	77.6%		MFS	48.3%		MFS	39.4%	

All nine classifiers have yielded higher average accuracies than the average MFS accuracy of the 150 ambiguous Punjabi noun words, which is 52.36%. The paired t-test, a statistical significance, test indicates that there exists a significant difference between the performance of nine ML classifiers using topical features (BoW and co-occurrence) and the baseline MFS method.

7.2.2 Analysis of the Punjabi WSD Based on Local Features

The experimented results of the training of nine supervised classifiers using two local features viz. collocation (one-word and two-word) and bigram having window size WS (2, 3 and 4) are shown in Table 7.3.

(i) The average predictive performance in Table 7.3 of nine classifiers using one-word and two-word collocation features is similar. That is, there is no significant difference between the performance of the classifier using one-word and two-word collocation features. Thus the increase in the collocation word combinations has no positive impact on the predictive performance of the classifiers. Table 7.3 shows that all the classifiers except the Stk and StkC have yielded better average accuracies when explored the one-word collocation feature set.

Table 7.3:Average accuracies of nine ML classifiers using collocation and bigram feature sets

Local Features		**Classifiers**									
		DL	DT	RF	NB	SVM	k-NN	AB	Stk	StkC	MFS
Collo-cations	One-word	76.24	76.90	80.34	80.87	80.91	71.19	79.12	80.34	81.29	52.36
	Two-word	74.98	76.41	78.02	80.67	80.64	65.76	79.04	80.92	**81.68**	
Bigram	WS2	73.71	73.67	78.55	77.92	78.28	64.90	75.62	77.46	**78.57**	
	WS3	59.65	59.76	66.78	65.55	65.19	55.39	60.17	63.46	64.52	
	WS4	55.94	55.92	60.22	59.40	54.30	54.79	55.82	57.37	58.05	

The one-word local feature set in the immediate vicinity of the target words provides an ample clue of its sense. The ensemble ML classifier StkC has outperformed the other ML classifiers for one-word and two-word collocation feature sets. The average accuracies of the ML classifiers using one-word and two-word collocation feature sets are pointedly higher than the baseline MSF accuracies.

(ii) The results of the nine ML classifiers using three bigram feature sets show that the bigram ws2 (immediate word combinations of target and its neighbouring words) are more suitable for the disambiguation process of the Punjabi language. There is a significant drop in the average accuracies of the ML classifiers using the bigram ws3 and ws4 feature sets. Figure 7.2 shows the average accuracies of the ML classifiers using three different bigram features. The yellow line in Figure 7.2 represents the baseline MFS. It further confirms that the words farther from the target ambiguous Punjabi word have a little or no impact on the sense classification of the target ambiguous word. It has been evident from Figure7.2 that the average accuracies produced by the ML classifiers using the bigram ws2 feature set are significantly higher than the baseline MSF. But the average accuracies of the ML classifiers based on the bigram ws3 and ws4 feature sets are maginaly high or similar to the average baseline MFS accuracy.

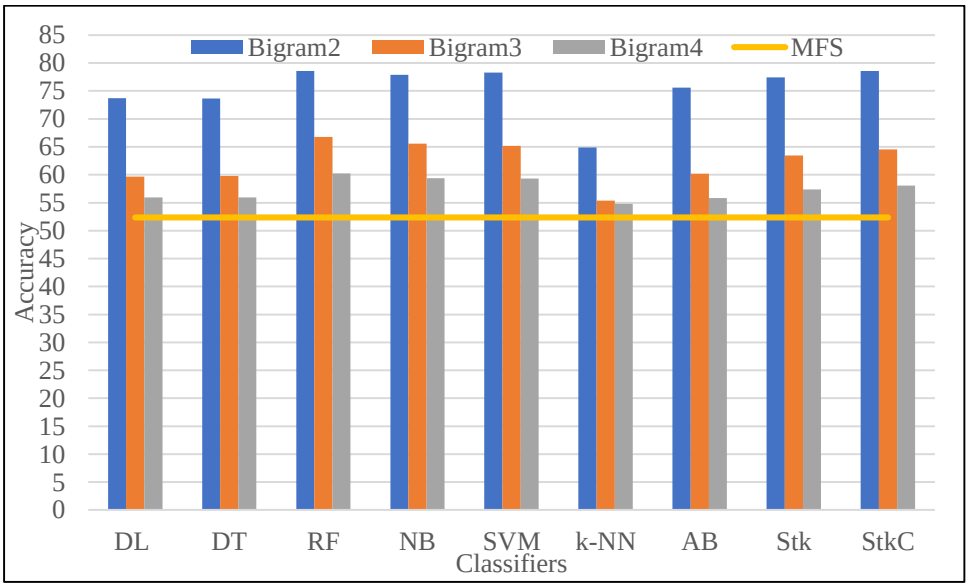


Figure 7.2: Average accuracies of the ML classifiers using bigram features

7.2.3 Analysis of the Punjabi WSD Based on Syntactic Features

The experimented results of the training of the nine supervised classifiers on five syntactic features namely *PoSw*, $PoSw_{+1}$ (PoS of target word *w* and the word immediate right *w+1* to it), PoS_{w-1} PoS_w (PoS of target word *w* and the word immediate left *w-1* to it), PoS_{w-1} PoS_{w+1} (PoS

of the words immediate right *w+1* and left to target word *w*), $PoS_{w-2}PoS_{w-1}\ PoS_w\ PoS_{w+1}\ PoS_{w+2}$ (PoS of the target word *w* and two words immediate right *w+1, w+2* and left *w-1, w-2* to it), $PoS_{w-1}PoS_w\ PoS_{w+1}$ (PoS of the target word *w* word and one word immediate right *w+1* and left *w-1* to it) are shown in Table 7.4. It can be observed from Table 7.4 that the average accuracies are not encouraging.

Table 7.4: Average accuracies of nine classifiers using PoS syntactic features

Syntactic Features (PoS)	**ML Classifiers**									
	DL	DT	RF	NB	SVM	k-NN	AB	Stk	StkC	MFS
PoS_w, PoS_{w+1}	56.77	51.61	51.61	52.25	53.54	48.38	54.58	54.78	56.78	52.36
$PoS_{w-1}\ PoS_w$	52.25	52.9	54.19	55.48	55.48	56.77	53.60	56.25	57.95	
$PoS_{w-1}\ PoS_{w+1}$	51.61	54.83	50.96	52.9	54.19	50.96	52.28	54.45	55.45	
$PoS_{w-2}PoS_{w-1}\ PoS_w\ PoS_{w+1}\ PoS_{w+2}$	50.96	54.19	50.96	53.54	52.25	54.83	55.15	55.85	57.85	
$PoS_{w-1}PoS_w\ PoS_{w+1}$	52.90	51.61	53.54	55.48	51.61	50.32	53.75	55.95	57.65	

For the $PoSw, PoSw_{+1}$ feature set, the StkC and DL classifiers have yielded the best average accuracies. The StkC and k-NN, as shown in Table 7.4, have outperformed other classifiers using the PoS combinations $PoS_{w-1}\ PoS_w$. While using the $PoS_{w-1}\ PoS_{w+1}$ feature set, the StkC and DT classifiers have produced the highest accuracy of 55.45% and 54.83%. For the $PoS_{w-2}PoS_{w-1}\ PoS_w\ PoS_{w+1}\ PoS_{w+2}$ feature set, the StkC and k-NN are the two top-performing classifiers. The StkC and Stk have produced the best average accuracies using the $PoS_{w-1}PoS_w\ PoS_{w+1}$ feature set. The best average accuracy of 57.95% has been resulted by the StkC classifier using the $PoS_{w-1}\ PoS_w$ feature set. However, it has been observed from Table 7.4 that the average accuracies produced by the nine ML classifiers are similar to the average MFS accuracy that is 53%. It has been further observed that the ensemble classifier StkC is the most successful classifier for the PoS feature sets. To determine the reason for the low accuracies by the ML classifiers, the PoS tagged Punjabi dataset of individual words has been manually analysed. It has been observed that the ambiguity of PoS tags for the Punjabi words is the major reason for the poor performance of the ML classifiers. It has been concluded that inadequate tagging by Punjabi shallow parser of the sense tagged dataset, and small syntactic combinations are the other major reasons for the poor results of the ML classifiers using syntactic PoS feature sets.

7.2.4 Analysis of the Punjabi WSD Based on Semantic Features

The average accuracy of the nine ML classifiers based on the word × document co-occurrence matrix is shown in Table 7.5. The term frequency-inverse document frequency (tf-idf) weights of individual words in the different sense tagged files of the Punjabi ambiguous word is estimated and incorporated into the word × document co-occurrence matrix. The average results using the word × document co-occurrence matrix have yielded a significantly enhanced predictive accuracy than the baseline MFS accuracy.

Table 7.5: Average accuracies of the ML classifiers using the word × document semantic feature

Semantic Feature	**ML Classifiers**									
	DL	DT	RF	NB	SVM	k-NN	AB	Stk	StkC	MFS
Word x document (Tf-idf weights)	77.74	78.45	77.87	85.48	81.40	69.74	81.67	87.24	88.40	52.36

The ensemble classifiers StkC and Stk have outperformed the other ML classifiers. However, the base ML classifier NB has also yielded a comparative average accuracy using the word × document semantic feature.

7.2.5 Analysis of the Word Embedding Features for ML Punjabi WSD

The average predictive performance of the ML classifiers using Word2Vec, Global Vector (GloVe) and FastText word embedding feature sets are discussed as follows.

(i) A series of experiments have been executed to study the impact of various parameters of Word2Vec word vectors of the Punjabi language on the performance of the Punjabi language disambiguation.

Table 7.6: Average accuracies of ML classifiers using Word2Vec word vectors and four combination techniques

Word2Vec Models	**Comb. methods**	**ML classifiers**									
		DL	DT	RF	NB	SVM	k-NN	AB	Stk	StkC	MFS
CBOW	Add	64.99	64.28	74.51	65.11	75.35	70.70	62.82	70.18	71.31	52.36
	Avg	65.18	64.46	75.33	75.26	77.28	72.91	70.51	73.21	74.15	
	FD	65.96	65.45	76.93	74.01	77.90	72.68	71.65	73.67	74.52	
	ED	67.15	66.70	76.73	68.42	75.90	71.98	71.89	72.01	72.96	
SG	Add	66.23	66.21	76.15	63.08	78.32	74.15	71.24	71.64	73.40	
	Avg	67.59	66.75	77.60	78.26	81.35	77.03	73.72	76.10	77.48	
	FD	68.96	68.00	79.53	80.14	**82.41**	78.43	75.11	77.92	79.29	
	ED	69.44	68.66	79.33	77.74	80.32	75.98	74.36	76.31	77.49	

The average accuracy produced by the nine ML classifiers using a continuous bag of word (CBoW) and SkipGram (SG) Word2Vec word vectors has been reported in Table 7.6.

The dense and reliable CBoW and SG word vectors have been combined using Addition (Add), Average (Avg), Fractional Decay (FD) and Exponential Decay (ED) techniques for the training of the traditional supervised ML classifiers. All classifiers have evaluated CBoW and SG word vectors on the 150 ambiguous Punjabi noun words independently. Table 7.6 shows the average accuracies produced by nine ML classifiers using CBoW and SG word vectors and four different combination techniques.

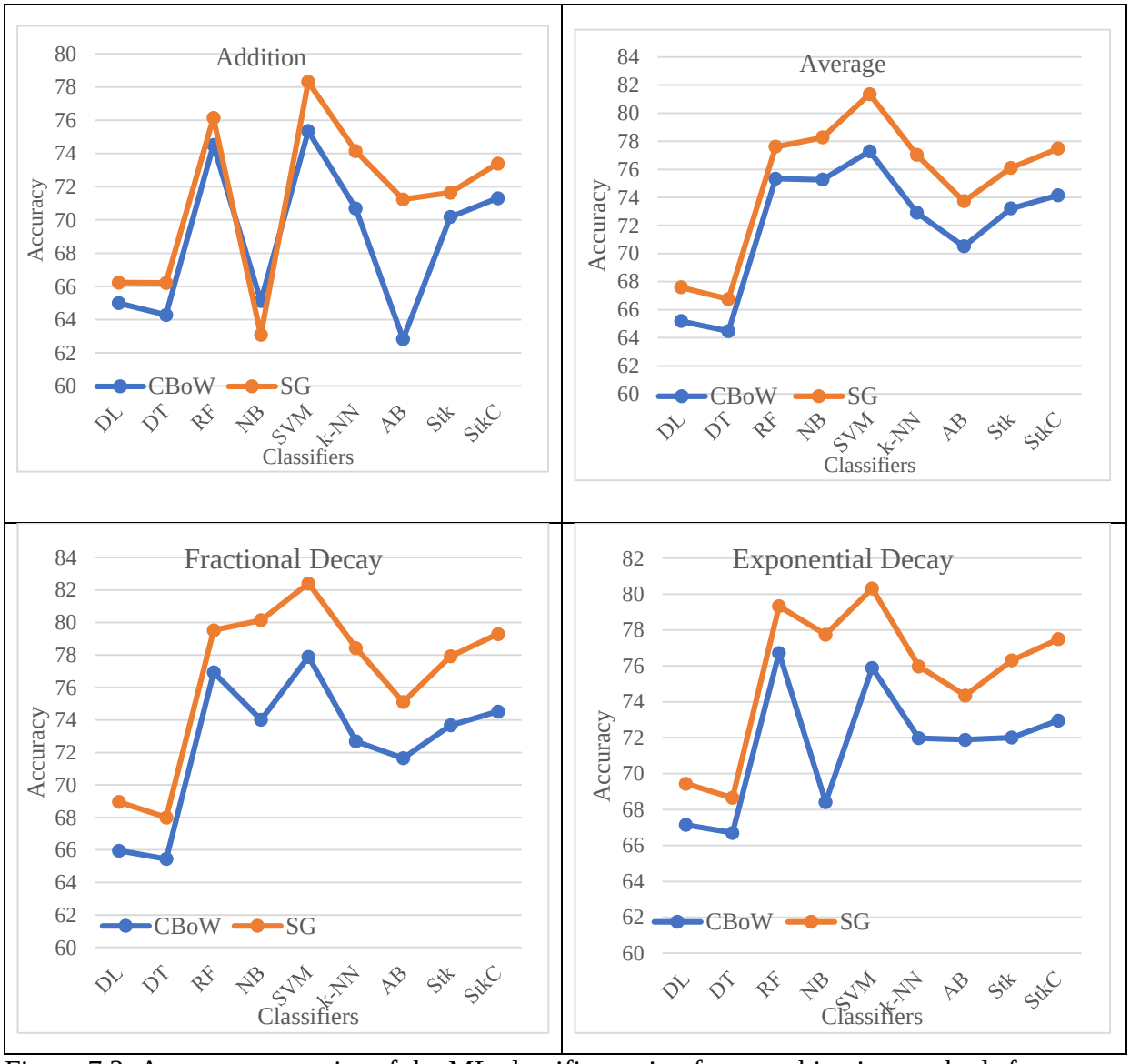


Figure 7.3: Average accuracies of the ML classifiers using four combination methods for CBoW and SG word vectors

Figure 7.3 depicts the average accuracy of nine ML classifiers which are trained using CBOW and SG word vectors. It is evident from Figure7.3 that SG Word2Vec vectors of

the Punjabi language have produced better average accuracies. The FD combination technique is the most suitable method to combine the CBoW and SG word vectors and to train the ML classifiers. The SVM classifier has produced the best average accuracies while using the SG word vectors. The predictive performance of the ensemble classifiers is lower than the base SVM classifiers. However, the performance of the nine ML classifiers is significantly better than the baseline MFS accuracy.

The analysis of accuracies obtained by the individual ML classifier using the best combination technique FD on the CBoW and SG word vectors is depicted in Figure7.4 and Figure7.5 respectively. The accuracies reported by DL and DT classifiers in Figure7.4 are significantly lower than those of the other seven ML classifiers.

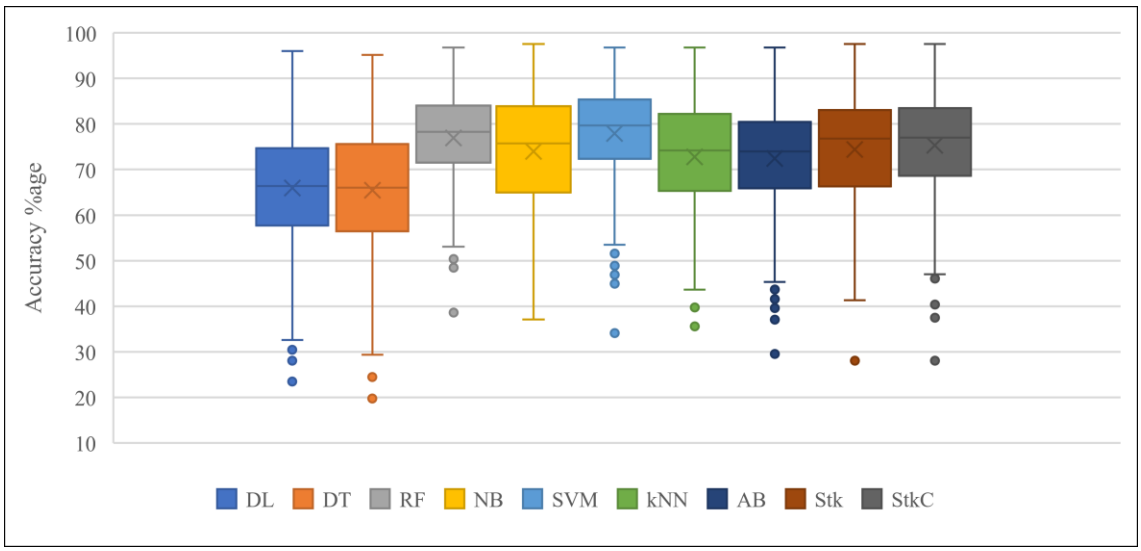


Figure 7.4: Distribution of accuraices yielded by ML classifiers using CBoW word vector with FD combination for Punjabi dataset

The interquartile ranges in Figure7.4 show that the accuracies reported by DL, DT and NB classifiers are widely dispersed. The smaller interquartile ranges of RF, SVM, AB and StkC in Figure indicate closely packed accuracies. The vast range of scores by DL, DT, NB classifiers advocates that their performance is highly dispersed on the Punjabi dataset. In contrast, the compact range of scores by RF, SVM and StkC classifiers suggests that their performance is comparable on the Punjabi dataset. Nine ML classifiers report the limited number of outliers out of 150 ambiguous Punjabi words. There is no outlier word for NB classifier. A single outlier word for Stk classifier. The kNN, DT classifiers have reported two outliers, DL and RF classifiers have reported three outliers, StkC has reported four

outliers. The highest number of outlier words, five, are reported for the SVM and AB classifiers.

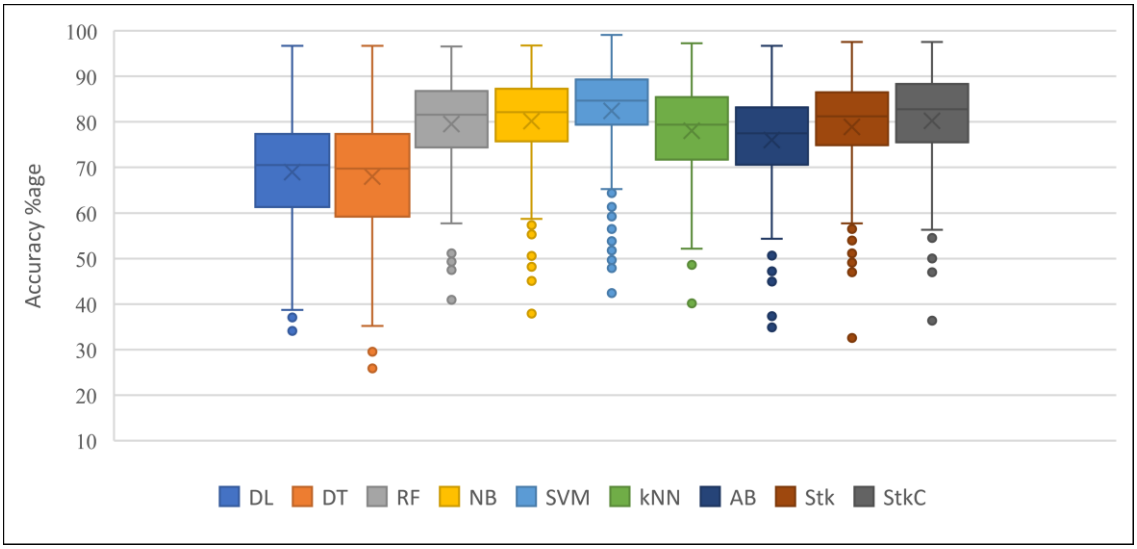


Figure 7.5: Distribution of accuraices yielded by ML classifiers using SG word vector with FD combination for Punjabi dataset

The accuracies reported by DL and DT classifiers in Figure7.5 are lower than the accuracies reported by the other seven ML classifiers when evaluated on the Punjabi dataset. The small interquartile ranges in Figure7.5 shows that the accuracies reported by all ML classifiers are closely packed. The large range of scores stated by DL, DT classifiers advocates that their performance is highly dispersed on the Punjabi dataset. In comparison, the compact range of scores stated by the other seven ML classifiers suggests that their performance is similar on the Punjabi dataset. The outliers reported by nine ML classifiers from 150 ambiguous Punjabi words in Figure7.5 are more than in Figure7.4 but countable.

(ii) The GloVe word vectors have been used to train the nine ML classifiers individually on 150 ambiguous Punjabi words, and the average accuracies of the ML classifiers are presented in Table 7.7.

Table 7.7: Average accuracies produced by the ML classifiers using GloVe word vectors

Comb. Methods	**ML classifiers**									
	DL	DT	RF	NB	SVM	k-NN	AB	Stk	StkC	MFS
ADD	60.52	59.60	70.79	55.50	72.77	67.27	66.34	65.71	68.08	52.36
AVG	72.68	72.83	**79.87**	66.52	65.71	72.29	77.75	74.99	76.03	
FD	63.18	62.71	74.29	68.26	76.98	70.67	70.67	71.56	72.89	
ED	56.90	56.94	60.63	53.02	57.36	57.40	58.25	62.26	64.23	

The four different combination techniques such as ADD, AVG, FD and ED have been used to combine the GloVe word vectors with ML classifiers to perform WSD. The RF classifier has yielded the best average predictive performance among the eight other ML classifiers using the FD combination technique.

(iii) The average accuracies produced by nine ML classifiers using the FastText word vector are presented in Table 7.8. It has been observed from Table 7.8 that the SVM classifier has yielded the best average accuracy using the FastText word vectors and FD combination technique.

Table 7.8: Average accuracies produced by the ML classifiers using FT word vectors

Comb. Methods	**ML classifiers**									
	DL	**DT**	**RF**	**NB**	**SVM**	**k-NN**	**AB**	**Stk**	**StkC**	**MFS**
ADD	64.02	62.96	74.14	59.00	81.18	72.00	71.15	71.63	75.03	52.36
AVG	60.75	60.68	71.87	75.14	77.38	71.69	64.94	71.86	73.62	
FD	67.81	63.29	79.41	81.15	**83.50**	77.91	75.77	79.97	81.68	
ED	68.60	68.22	79.12	79.94	81.37	76.14	75.58	78.84	79.87	

The evaluation of three different word representations indicates that the FastText word vectors are the most suitable Punjabi word embedding feature set among the Word2Vec, GloVe and FastText vector representations. It has been observed that the SVM classifier has yielded the best average accuracy using the Word2Vec, and FastText vector representations. The advantages of word embedding features are that they are dense in nature, scalable and provides competitive performance when combined with ML classifiers. The performance of these Neural Network (NN) based word vectors can be further improved by tuning different parameters of the Word2Vec, GloVe and FastText algorithms.

7.3 Results of the Punjabi WSD Using Deep Learning Classifiers

Two deep learning classifiers, the Multi Layer Perceptron (MLP) and Long Short Term Memory (LSTM), have been individually trained on the word index vectors (selected from the sense tagged dataset of the 150 ambiguous Punjabi word) and on three different NN based word vectors Word2Vec (CBoW, SG), GloVe and FastText (prepared using raw Punjabi corpus). The 80:20 split validation approach has been applied for the training of the NN based MLP and LSTM classifiers. The average accuracies produced by MLP and LSTM deep learning classifiers using word index vectors are shown in Table 7.9.

Table 7.9:Average accuracies of deep learning classifiers using word index vector

Feature	**Deep Learning Classifiers**		**MFS**
Word index vector	MLP (82.35)	LSTM (83.78)	52.36

It has been observed that the average predictive performance of the MLP and LSTM deep learning classifiers using word index vector is significantly higher that the average MFS accuracies. However, the word index vector is a binary vector and is selected from the limited sense tagged dataset.

The MLP and LSTM deep learning classifiers have directly trained on the dense word vectors. The average predictive performance of the MLP and LSTM deep learning classifiers based on the NN based word vectors Word2Vec (CBoW, SG), GloVe and FT are shown in Table 7.10. During the training of MLP and LSTM deep learning classifiers, optimal values of parameters such as hidden layers, activation function, and loss function are manually selected by applying heuristics.

Table 7.10:Average accuracies of deep learning classifiers using word embedding vectors

Features	**Deep Learning Classifiers**		**MFS**
	MLP	LSTM	
Word2Vec CBoW	82.45	82.96	52.36
Word2Vec SG	83.15	**84.15**	
GloVe	78.26	80.15	
FastText	79.15	80.85	

It has been observed that the word vector features produced by the Word2Vec SG model are more suitable for the small dataset of the Punjabi language. The LSTM classifier has yielded the best performance (84.15%) using the Word2Vec SG word embedding. The paired t-test, a statistical significance test, indicates that there exists a significant difference between the performance of the MLP and LSTM classifiers using word vector features and the baseline MFS accuracy.

7.4 Testing of the Punjabi WSD System

The trained Punjabi WSD system has tested on a small Punjabi test dataset. The test dataset has been manually prepared for the individual Punjabi ambiguous words such that the test instances are not present in the training dataset. On average, the test dataset of the independent target word is 15% of the instance count of its training dataset. The trained deep learning classifiers of individual Punjabi word determines the senses of the instances of individual

Punjabi ambiguous word. The senses estimated by the Punjabi WSD system have been manually checked. The average accuracies obtained by MLP and LSTM classifiers from the Punjabi test dataset using the Word2Vec SG word embedding features are presented in Table 7.11. The average accuracies indicate that the MLP and LSTM outperform the baseline MFS accuracy.

Table 7.11: Predictive performance of the MLP and LSTM on the test dataset

Classifiers	**MLP**	**LSTM**	**MFS**
Average Accuracy	77.24	71.5	52.36

The trained ML classifiers have also predicted the senses for the instances of the 150 ambiguous Punjabi noun words from the unseen test dataset. The average accuracies yielded by the trained ML classifiers on the BoW feature set are shown in Table 7.12. It has been observed from Table 7.12 that the average accuracies of the ML classifiers are also significantly higher than the MFS. However, these accuracies are less than the accuracies generated by the trained ML models (using cross-validation) or deep learning models (using split validation).

Table 7.12:Average accuracies produced by the ML classifiers on the test dataset

ML Classifiers	**DL**	**DT**	**RF**	**NB**	**SVM**	**k-NN**	**AB**	**Stk**	**StkC**	**MFS**
Average Accuracies	69.75	70.75	75.28	**79.45**	76.65	61.70	71.35	76.80	78.45	52.36

7.5 Error Analysis of the Punjabi WSD System

In this section, the error analysis of the Punjabi WSD system based on machine learning and deep learning classifiers have been discussed.

7.5.1 Error Analysis of the ML Punjabi WSD System

The major reasons for the misclassification of ambiguous Punjabi noun words from the Punjabi instances by the supervised Punjabi WSD system are the sparse sense tagged dataset, high polysemy, skewed sense tagged dataset and optimal feature selection. Firstly, it has been observed that the Punjabi WSD system yields better results for the ambiguous words having a large tagged dataset. Thus, the Punjabi WSD system error can be reduced by enhancing the size of the sense tagged dataset.

Secondly, the high polysemy that is the words having a large number of senses is another reason for degradation in the performance of the Punjabi WSD system. A large number of senses of the ambiguous word have a small size of the sense tagged dataset that is the rare senses of ambiguous Punjab have negligible Punjabi instances. Therefore, the Punjabi WSD system fails

to identify the rare senses of ambiguous Punjabi noun word accurately. The sense tagged dataset is skewed towards the most frequent sense of the ambiguous word. Consequently, the Punjabi WSD system generally assigns the most frequent sense of the ambiguous word.

Finally, the small size of the context of the Punjabi sense tagged dataset causes the selection of overlapping local features, and thus the Punjabi WSD system misclassifies the sense of the ambiguous word. Therefore, enriching the context size of the sense tagged dataset can reduce the error of the Punjabi WSD system. The accuracy of the Punjabi WSD system using syntactic features is depended on the correctness of the Punjabi PoS tagged dataset. The correctness of the Punjabi PoS tagged dataset is dependent on the Punjabi PoS tagger. If the PoS tagger does not identify the correct PoS tags, then the syntactic features will not be correctly prepared by the proposed system. The errors of the ML classifiers using the word embedding feature sets are due to the missing of word vectors for the Punjabi words of sense tagged dataset.

Besides the problems of dataset sparseness, skewness and overlapping of senses, the Punjabi WSD system also has limited coverage. It can disambiguate only the targeted 150 ambiguous Punjabi noun words. The Punjabi WSD system cannot handle the PoS classes such as verbs, adverbs and adjectives. The Punjabi WSD system also lacks in the handling of the Punjabi multiword. However, the Punjabi WSD system using ML classifiers can facilitate the users to disambiguate the ambiguous words correctly.

7.5.2 Error Analysis of the Deep Learning Punjabi WSD System

The analysis of the result of the MLP and LSTM deep learning classifiers indicates that the size and quality of the sense tagged dataset and raw corpus dataset of the Punjabi language are the major reasons for the misclassification of senses of the ambiguous words. The predictive performance of the MLP and LSTM deep learning classifiers using the word index feature vectors crafted from the sense tagged dataset is hindered due to the polysemy and data skewness.

However, the deep learning Punjabi WSD system using the word embedding features extracted from the raw corpus has an advantage over the ML Punjabi WSD system. The MLP and LSTM classifiers using the word embedding features overcome the feature engineering process and thus limits the manual intervention in the disambiguation process. The predictive performance of the MLP and LSTM classifiers using the word embedding features is dependent on the quality of the word embedding vectors. The Punjabi word embedding features have been prepared from the relatively small Punjabi raw corpus. Therefore, the MLP and LSTM

classifiers have also caused an error in sense disambiguation. Besides the issues like small corpus and overlapping senses of the ambiguous words, the Punjabi WSD system using deep learning classifiers needs many computational resources and a large amount of time.

7.6 Comparison of Punjabi WSD with Existing Systems

The Punjabi WSD developed and investigated in this research work has been compared with the existing Punjabi WSD approaches. The comparison of differnet Punjabi WSD system gives an insight into the sense inventory applied by different WSD systems, corpus used by the systems, techniques employeed and evaluation metrics explored by the researchers for Punjabi sense disambiguation.

The earlier work of the Punjabi WSD has been reported in chapter2 of this thesis. It indicates that the different researchers have explored different classification approaches to disambiguate the Punjabi words. Initially, researchers have explored the knowledge base Lesk algorithm for the Punjab WSD. Recently the supervised classifiers NB, k-NN, DT, DL and SVM have applied for the Punjabi WSD. Punjabi WordNet has been used by the researcher as the Punjabi knowledge resource. Most of the researchers have evaluated their respective systems using accuracy or recall metrics. However, researchers have explored their different classification techniques on different Punjabi words.

Table 7.13:List of the Punjabi WSD systems

WSD System	**Knowledge Resource**	**Sense tagged Corpus**	**Number of ambiguous words**	**Techniques**	**Evaluation metric**
(Rana & Kumar, 2015)	Punjabi WordNet	Not specified	30	Overlap Lesk	Accuracy
(Kaur & Singh, 2015)	Not specified	WordNet	10	Modified Overlap Lesk	Recall
(Singh & Singh, 2015)	Punjabi WordNet	1549 instances	15	Modified Overlap Lesk	Precision
(Walia, et al., 2017)	Not specified	Evaluations and Language Resources Distribution Agency (ELRDA)	3	NB	Accuracy
(Walia, et al., 2018)	Not specified	ELRDA	8	k-NN	Accuracy
(Walia, et al., 2018)	Punjabi WordNet	ELRDA	20	NB, k-NN and DT	Accuracy
Our Punjabi WSD	Punjabi WordNet	22153 instances	150	NB, k-NN, DT, DL, SVM, AB,Stk,StkC,MLP and LSTM	Accuracy

Thus the non-availability of a standard dataset makes comparison among the different studies a trivial task. Table 7.13 shows the knowledge base used, sense tagged corpus size, the number of ambiguous words, techniques and evaluation metrics parameters used in different studies. From Table 7.13 it is concluded that our Punjabi WSD system has been extensively trained and tested on the 150 Punjabi ambiguous noun words. We have created a sense tagged dataset having 22153 instances for 150 Punjabi ambiguous words. Nine supervised classifiers and two deep learning classifiers have trained and tested on topical, local, syntactic, semantic Word2Vec, GloVe and FastText embedding features of 150 Punjabi ambiguous words.

This supervised Punjabi WSD system has yielded exceedingly good accuracies. However, the major limitation of the supervised Punjabi WSD is scalability. It requires sense tagged dataset for every new Punjabi ambiguous word. The machine learning and deep learning classifiers must be trained for the new Punjabi ambiguous words which, is a time consuming and a redundant task.

Chapter Summary

In this chapter, the predictive performance of the Punjabi WSD has been discussed. The accuracy metric yielded by the machine and deep learning classifiers has been compared with the baseline MFS accuracy. The results clearly indicate that predictive accuracies by machine and deep learning classifiers are higher than the baseline MFS accuracy for all the 150 ambiguous words while using topical, local and semantic features. The results show that the high polysemy, sparseness and skewed datasets reduce the performance of the supervised Punjabi WSD. The five syntactic features (five different PoS combinations) have not yielded encouraging results for the disambiguation of the ambiguous Punjabi nouns.

The ML classifiers have evaluated the predictive Word2Vec (CBoW, SG), GloVe, FastText word embedding features for the Punjabi WSD. The four combination techniques, addition, average, fractional and exponential decay, have been implemented to combine word vectors into ML classifiers. The predictive word embedding representations have produced encouraging results when the MLP and LSTM deep learning classifiers have applied. The word representations need minimum manual interventions for their generation. But the deep learning classifiers and the word vector representations generation are computational intensive activities. The different Punjabi and Indian language WSD systems have compared with our supervised WSD system. Our supervised Punjabi WSD has outperformed the other knowledge based and unsupervised Punjabi WSD systems.

Chapter 8 : Conclusions and Future scope

The Punjabi Word Sense Disambiguation (WSD) identifies the meaning of the ambiguous Punjabi word in the context. In this research work, the supervised Punjabi WSD has developed and experimented on 150 ambiguous Punjabi noun words selected from the Punjabi WordNet. The sense tagged dataset for the Punjabi language has prepared in a short period and with the limited intervention of linguistic experts. The Punjabi WSD system has produced very encouraging results. Therefore, the Punjabi WSD system can be helpful for other natural language processing tasks which in turn will benefit society. In this chapter, the conclusions and future implications of this research work have presented.

8.1 Conclusions

The detailed literature review of the WSD has shown that there exists little work for the Punjabi WSD. The Punjabi WSD is primarily used as a vivo module (as a part of the application) in the Punjabi machine translation tasks. The researchers have explored the knowledge based WSD approaches for the Punjabi WSD. The Punjabi language is an essential and widely used language of the Indo-Aryan language family, and it has important deviating features from the English language. It has been observed from this study that the unique characteristics of the Punjabi language need special attention to perform the Punjabi WSD.

The WSD task is dependent on the selection of ambiguous words and their respective senses. The central pivot of this research is the Punjabi WordNet. It has acted as a primary knowledge resource of the Punjabi WSD. The 150 most ambiguous Punjabi noun words and their senses have been selected from the Punjabi WordNet. It has been observed that the sense definitions (fine sense or coarse sense) for the WSD are task-dependent. Primarily, the coarse senses have been applied for the WSD task. In this research, the fine senses of the ambiguous words extracted from the Punjabi WordNet are converged into the coarse senses. A set of rules have been devised that merge the similar fine senses into more discern coarse senses of the 150 Punjabi ambiguous words. The dataset for the coarse senses of the ambiguous words has been prepared manually by selecting the text of ambiguous words from the online and offline Punjabi resources. The sense tagged dataset prepared in this work is skewed. The multiple annotators have validated the collected dataset of the individual word using the kappa statistics. The kappa statistics show a substantial agreement among the different annotators for the collected, and labelled dataset of the individual word.

The architecture of the Punjabi WSD has designed to implement automatic Punjabi language disambiguation. This architecture has been divided into the data collection, pre-processing, parsing, feature selection, feature vector generation, and classifier training phases to perform the disambiguation of ambiguous Punjabi words. Each phase of the Punjabi WSD architecture is independent of other phases. However, this modular approach of the Punjabi WSD architecture has helped us in design, develop, and debug each phase efficiently. This Punjabi WSD architecture has been designed and developed for supervised machine learning and deep learning classifiers. This architecture has experimented individually on the 150 ambiguous Punjabi noun words.

The pre-processing phase of the Punjabi WSD has processed the context of the labelled dataset to eliminate unnecessary noise from it. The punctuations, acronyms and wrong spelling have been removed with the help of the dictionaries. The Punjabi abbreviation completion dictionary, a standard Punjabi spelling dictionary and Punjabi stopwords dictionary are the outcomes of the pre-processing phase of the architecture of the Punjabi WSD. The parsing phase has used the Punjabi shallow parser to generate the part of speech tagged Punjabi dataset from the sense tagged dataset. The topical, local, syntactic and semantic features have selected from the pre-processed Punjabi text using different feature selection algorithms. The feature selection has produced individual feature vector files for 150 ambiguous words. These text features have been converted into binary and weighted vectors. To generate the Punjabi word embedding features of different dimensions from the unlabelled Punjabi corpus (collected from Punjabi Wikipedia and other Punjabi online corpora), the neural network based Word2Vec, Global Vector (GloVe), and FastText methods have applied. The word embedding features have directly produced the weight feature vectors of fixed dimension for the Punjabi words. Therefore, these word embedding features need no further processing.

The Naïve Bayes, Decision List, Decision tree, k-Nearest Neighbour, Support Vector Machine (SVM), AdaBoost, Stacking and StackingC machine learning and Multi Layer Perceptron (MLP) and Long Short Term Memory (LSTM) deep learning classification techniques in conjunction with an empirical analysis for five statistical and prediction feature extraction methods have evaluated individually on the Punjabi noun words. The machine learning classifiers of the WEKA toolkit have trained using a 10-fold cross-validation approach. The classifiers have produced the best results using the bag of words feature. Ensemble classifiers Stacking and StackingC have produced the best results for the multiclass ambiguous words. The deep learning classifiers MLP and LSTM using word embedding features have yielded

better results than simple word index features. The machine learning classifiers have also experimented on the word embedding features. The SVM classifier has produced a comparable performance with deep learning classifiers using the word embedding features. The pre-processing of the text while selecting lexical features has improved the performance of some classifiers.

The accuracy metric has been used to measure the performance of the explicit Punjabi WSD system for the individual ambiguous word. The estimated accuracies of the individual word using supervised classifiers on lexical and sematic feature sets are better than the baseline accuracy (most frequent sense). The accuracy of the classifiers is inadequate when the syntactic features have used. It has been observed that syntactic tagging is dependent on the Part of Speech (PoS) tagger. The analysis of the PoS tags of the Punjabi text shows the high error rate in the PoS tagging. The results of this experiment show that the words having more senses are difficult to disambiguate. The words having an enormous sense labelled dataset produce better predictions for the unknown instances. The words having skewed dataset are prone to the most frequent sense assignment anomaly. The ensemble classifiers produce better performance for the skewed dataset of the Punjabi noun words. The supervised Punjabi WSD has yielded significantly higher accuracy than the existing knowledge base classifiers of the Punjabi WSD.

A web interface has been prepared using the java server pages and python scripting language for the supervised targeted Punjabi WSD and is available at http://www.punjabiwsd.in/. The ambiguous Punjabi words, their respective coarse senses, and the example instances are stored in a simple JavaScript Object Notation (JSON) data structure. The trained models for the individual words using different feature sets have been prepared and stored on the server. These trained models are used to identify the words in the test instances supplied by the user. This system identifies the ambiguous words of the test instance present in a sense tagged dataset.

8.2 Future Scope

Several aspects should be beneficial to extend the targeted Punjabi WSD for the different Punjabi Natural Langauge Processing (NLP) tasks in the future.

(i) The sense annotated corpus of the Punjabi language created in this work is available to the researchers. The researchers can explore new approaches for the WSD for the Punjabi language.

(ii) To overcome the knowledge bottleneck, the corpus can be improved and expanded through crowdfunding.

(iii) In the future, the sense tagged dataset would be scaled further by including more ambiguous Punjabi nouns, verbs, adverbs, and adjective words. This enhancement will help the researchers to explores the all word WSD of the Punjabi language. The all word Punjabi WSD will have better coverage of the language.

(iv) The concertation phase will be incorporated during the sense annotation process. In this phase, the instances on which the annotators and evaluators do not have sense agreement will not be rejected. Nevertheless, the annotators and evaluators will try to reach a consensus to include the disagreed instances in the dataset. The concertation phase will eliminate the bias in future and will enrich the dataset for the ambiguous Punjabi words.

(v) The sense labelled dataset can be used to prepare a parallel dataset among the other Indian languages, which can be used to explore new approaches of the Punjabi WSD.

(vi) The sense tagged dataset of the Punjabi language will be useful in obtaining an objective of a comparative study of the performance of the Punjabi WSD systems between the participant teams in evaluation competitions like the Association for Computational Linguistics, and SemEval, *etc*.

(vii) The syntactic features of the Punjabi language further need to be explored. There is an urgent requirement of quality linguistic resources such as the shallow parser, PoS tagger for the Punjabi language. The improved linguistic resources will enhance the quality of the syntactic features, and accordingly, the performance of the WSD tasks will improve.

(viii) The prepared distributional features using Word2Vec, GloVe, and FastText neural network based word vectors of the Punjabi language can be explored by researchers for different NLP tasks.

(ix) In the future, the different language models will be explored for the Punjabi WSD. The latest state-of-the-art transformer models for the English language WSD can be adapted for WSD of the Punjabi language. The IndicNLPSuite will be investigated for the Punjabi WSD.

(x) In the future, the accuracy of the Punjabi WSD system can be improved by the re-training of the models. The WSD system in the re-training approach will include the predicted dataset in the training dataset. The re-training approach will increase the size

of the training dataset. That can further improve the accuracy of the Punjabi WSD system.

(xi) The performance of the ensemble classifiers can be improved by including distributional architecture for the training of the individual base classifiers.

(xii) The full-fledged applications should be built containing the Punjabi WSD either as an integrated or a pluggable component.

(xiii) To integrate this explicit Punjabi WSD system with the other NLP tasks, APIs would be prepared. In the future, this integration would enhance the performance of the other challenges such as machine translation, sarcasm detection, sentiment analysis, question answering of the Punjabi language. The proposed Punjabi WSD system can be used for the benefit of society.

(xiv) The Punjabi WSD system should not stay as an independent application. It should be incorporated and evaluated as a component in the different NLP applications like information extraction, information retrieval, machine translation, semantic Web, *etc*. The proposed Punjabi WSD system's performance is encouraging and can provide impetus to the development and improvement of the different NLP tasks of the Punjabi language.

List of Publications by the Author

Papers in refereed journals

1. Varinder pal Singh and Parteek Kumar, Naive Bayes Classifier for Word Sense Disambiguation of Punjabi Language. Malaysian Journal of Computer Science, 31(3), pp. 188–199, 2018. https://doi.org/10.22452/mjcs.vol31no3.2
2. Varinder pal Singh and Parteek Kumar, Sense disambiguation for Punjabi language using supervised machine learning techniques. Sādhanā, 44(11), pp. 1-15, 2019. https://www.ias.ac.in/article/fulltext/sadh/044/11/0226
3. Varinder pal Singh and Parteek Kumar, Word sense disambiguation for Punjabi language using deep learning techniques. Neural Computing and Applications 32, pp. 2963–2973 (2020). https://doi.org/10.1007/s00521-019-04581-3

References

1. Abid, M., Habib, A., Ashraf, J., & Shahid, A. (2018). Urdu Word Sense Disambiguation Using Machine Learning Approach. *Cluster Comput, 21*, 515-522. doi: https://doi.org/10.1007/s10586-017-0918-0

2. Agirre, E., De Lacalle, O. L., & Soroa, A. (2009). Knowledge-based WSD on Specific Domains: Performing Better Than Generic Supervised WSD. *Twenty-First International Joint Conference on Artificial Intelligence* (pp. 1501-1506). Pasadena,California: Morgan Kaufmann Publishers Inc.

3. Agirre, E., Lopez de Lacalle, O., & Soroa, A. (2014). Random Walks for Knowledge-based Word Sense Disambiguation. *Computational Linguistics, 40*(1), 57-84.

4. Agirre, E., Martinez, D., de Lacalle, O. L., & Soroa, A. (2006). Two Graph-based Algorithms for State-of-the-Art WSD. *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing* (pp. 585-593). Sydney, Australia: Association for Computational Linguistics.

5. Agirre, E., Soroa, A., & Stevenson, M. (2010). Graph-based Word Sense Disambiguation of Biomedical Documents. *Bioinformatics, 26*(22), 2889-2896.

6. Akkaya, C., Wiebe, J., & Mihalcea, R. (2009). Subjectivity word sense disambiguation. *Proceedings of the 2009 conference on empirical methods in natural language processing* (pp. 190-199). Singapore: Association for Computational Linguistics.

7. Amancio, D. R., Comin, C. H., Casanova, D., Travieso, G., Bruno, O. M., Rodrigues, F. A., & da Fontoura Costa, L. (2014). A Systematic Comparison of Supervised Classifiers. *PloS one, 9*(4), 1-14.

8. Anand Kumar, M., Rajendran, S., & Soman, K. P. (2014). Tamil Word Sense Disambiguation Using Support Vector Machines with Rich Features. *International Journal of Applied Engineering Research, 9*(20), 7609-7620.

9. Arif, Z., Yaqoob, M., Rehman, A., & Jamil, F. (2016). Word Sense Disambiguation for Urdu Text by Machine Learning. *International Journal of Computer Science and Information Security (IJCSIS), 14*(5), 738-757.

10. Athaiya, A., Modi, D., & Pareek, G. (2018). A Genetic Algorithm Based Approach for Hindi Word Sense Disambiguation. (pp. 11-14). Coimbatore, India: IEEE.

11. Azzini, A., Pereira, C. d., Dragoni, M., & Tettamanzi, A. G. (2008). Evolving Neural Networks for Word Sense Disambiguation. *2008 Eighth International Conference on Hybrid Intelligent Systems* (pp. 332-337). Barcelona: IEEE. doi:doi: 10.1109/HIS.2008.88

12. Bala, P. (2013). Knowledge Based Approach for Word Sense Disambiguation using Hindi WordNet. *The International Journal Of Engineering And Science (IJES),* 36-41.

13. Bala, P. (2013). Word Sense Disambiguation Using Selectional Restriction. *International Journal of Scientific and Research Publications, 3*(4), 1-4.

14. Banerjee, S., & Pedersen, T. (2003). Extended Gloss Overlaps As a Measure of Semantic Relatedness. *In Proceedings of the 18th International Joint Conference on Artificial Intelligence* (pp. 805-810). Acapulco, Mexico: Morgan Kaufmann Publishers Inc.

15. Basile, P., Caputo, A., & Semeraro, G. (2014). An Enhanced Lesk Word Sense Disambiguation Algorithm through a Distributional Semantic Model. *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, (pp. 1591-1600).

16. Baskaran, S., & Vaidehi, V. (2004). Collocation based word sense disambiguation using clustering for Tamil.

17. Basu, C., Dietz, L., & Fellbaum, C. (2018). WordN*et*Context: Information Retrieval-friendly Access to WordNet Senses. *In ProfS/KG4IR/Data: Search@ SIGIR*, (pp. 63-64).

18. Bellegarda, J. R., & Barman, B. (2019). *USA Patent No. 10,311,144.*

19. Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A Neural Probabilistic Language Model. *Journal of machine learning research, 3*, 1137--1155.

20. Bernard, J. R. (1986). *Macquarie Thesaurus. Macquarie.* Sydney, Australia.

21. Bevilacqua, M., & Navigli, R. (2019). Quasi Bidirectional Encoder Representations from Transformers for Word Sense Disambiguation. (pp. 122-133). Varna, Bulgaria: INCOMA Ltd.

22. Bhala, R. V., & Abirami, S. (2014). Trends in Word Sense Disambiguation. *Artificial Intelligence Review, 42*(2), 159--171.

23. Bharati, A., Rao, K. P., Sangal, R., & Bendre, S. M. (2000). Basic Statistical Analysis of Corpus and Cross Comparison Among Corpora. *Proceedings of the International Conference on Natural Language Processing (ICON 2002).* Mumbai.

24. Bhattacharya, P. (2010). IndoWordNet. *Proceeding of Lexical Resources Engineering* (pp. 1-8). Malta: European Language Resources Association (ELRA).

25. Bhattacharyya, P. (2017). IndoWordNet. In N. S. Dash, P. Bhattacharyya, & J. D. Pawar (Eds.), *The WordNet in Indian Languages* (pp. 1-18). Singapore: Springer.

26. Bhingardive, S., & Bhattacharyya, P. (2017). Word Sense Disambiguation Using IndoWordNet. In N. S. Dash, P. Bhattacharyya, & J. D. Pawar (Eds.), *The WordNet in Indian Languages* (pp. 243-260). Springer.

27. Bhingardive, S., Puduppully, R., Singh, D., & Bhattacharyya, P. (2014). Merging Verb Senses of Hindi Wordnet using Word Embeddings. *Proceedings of the 11th International Conference on Natural Language Processing* (pp. 344-352). Goa, India: NLP Association of India.

28. Bhingardive, S., Singh, D., Rudramurthy, V., Redkar, H., & Bhattacharyya, P. (2015). Unsupervised Most Frequent Sense Detection Using Word Embeddings. *Proceedings of the 2015 conference of the North American Chapter of the Association for Computational Linguistics: Human language technologies* (pp. 1238-1243). Denver, Colorado: Association for Computational Linguistics.

29. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics, 5*, 135-146.

30. Bond, F., & Foster, R. (2013). Linking and Extending an Open Multilingual Wordnet. (pp. 1352-1362). Sofia, Bulgaria: Association for Computational Linguistics.

31. Borah, P. P., Talukdar, G., & Baruah, A. (2014). Assamese Word Sense Disambiguation using Supervised Learning. *2014 International Conference on Contemporary Computing and Informatics* (pp. 946-950). IEEE.

32. Borah, P. P., Talukdar, G., & Baruah, A. (2019). WSD for Assamese Language. *Recent Developments in Machine Learning and Data Analytics. Advances in Intelligent Systems and Computing* (pp. 119-128). Singapore: Springer.

33. Brody, S., Navigli, R., & Lapata, M. (2006). Ensemble Methods for Unsupervised WSD. (pp. 97-104). Sydney, Australia: Association for Computational Linguistics.

34. Butnaru, A., Ionescu, R., & Hristea, F. (2017). ShotgunWSD: An UnsupervisedAalgorithm for Global Word Sense Disambiguation Inspired by DNA Sequencing. (pp. 916-926). Valencia, Spain: Association for Computational Linguistics.

35. Cai, J. F., Lee, W. S., & Teh, Y. W. (2007). NUS-ML: Improving Word Sense Disambiguation Using Topic Features. Prague, Czech Republic: Association for Computational Linguistics.

36. Chakrabarty, A., & Garain, U. (2016). BenLem (A Bengali Lemmatizer) and Its Role in WSD. *ACM Trans. Asian Low-Resour. Lang. Inf. Process., 15*(3), 1-18.

37. Chakrawarti, R. K., Bansal, P., & Bansal, J. (2020). Phrase-Based Statistical Machine Translation of Hindi Poetries into English by incorporating Word Sense Disambiguation. *SAMRIDDHI, 12*(1), 55-61.

38. Chand, S. (2016). Retrieved 06 06, 2020, from http://www.yourarticlelibrary.com/language/indian-languages-classification-of-indian-languages/19813/

39. Chaplot, D., Bhattacharyya, P., & Paranjape, A. (2015). Unsupervised Word Sense Disambiguation using Markov Random Field and Dependency Parser. *Twenty-Ninth AAAI Conference on Artificial Intelligence* (pp. 2217-2223). Austin, Texas: AAAI Press.

40. Chatterjee, A., Joshi, S. R., Khapra, M. M., & Bhattacharyya, P. (2010). Introduction to Tools for IndoWordNet and Word Sense Disambiguation. *In 3rd IndoWordNet workshop, International Conference on Natural Language Procesing* (pp. 1-7). Mumbai: IIT Bombay.

41. Chen, P., Bowes, C., Ding, W., & Choly, M. (2010). Word Sense Disambiguation with Automatically Acquired Knowledge. *IEEE Intelligent Systems, 27*(4), 46-55.

42. Chifu, A., & Hristea, F. (2018). Feature Selection for Spectral Clustering: to Help or not to Help Spectral Clustering When Performing Sense Discrimination for IR? *Open Computer Science, 8*(1), 218-227.

43. Chollet, F. (2015). Keras. GitHub. Retrieved from https://github.com/fchollet/keras

44. Ciaramita, M., & Altun, Y. (2006). Broad-coverage Sense Disambiguation and Information Extraction with a Supersense Sequence Tagger. *Conference on Empirical Methods in Natural Language Processing* (pp. 594-602). Sydney, Australia: Association for Computing Machinery.

45. Clement, J. (2020). *Leading countries based on Facebook audience size as of July 2020.* Retrieved July 2020, from https://www.statista.com/statistics/268136/top-15-countries-based-on-number-of-facebook-users/

46. Collobert, R., & Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. *Proceedings of the 25th international conference on Machine learning* (pp. 160–167). Helsinki, Finland: Association for Computing Machinery.

47. Dandala, B., Mihalcea, R., & Bunescu, R. (2013). Word Sense Disambiguation Using Wikipedia. In I. Gurevych, & J. Kim (Eds.), *The People's Web Meets NLP* (pp. 241-262). Berlin, Heidelberg: Springer.

48. Das Dawn, D., Shaikh, S. H., & Pal, R. K. (2020). A Comprehensive Review of Bengali Word Sense Disambiguation. *Artificial Intelligence Review, 53,* 4183-4213.

49. Das, A., & Sarkar, S. (2013). Word Sense Disambiguation in Bengali applied to Bengali-Hindi Machine Translation. *International conference on natural language processing. 10,* pp. 20-28. Noida India: ICON.

50. Decadt, B., Hoste, V., Daelemans, W., & Van Den Bosch, A. (2004). GAMBL, Genetic Algorithm Optimization of Memory-based WSD. (pp. 108-112). Barcelona, Spain: Association of Computational Linguistic.

51. Deng, L., & Yu, D. (2014). Deep Learning: Methods and Applications. *Foundations and trends in signal processing, 7*(3-4), 197--387.

52. Desai, B., Naik, K., & Bhatt, P. (2015). Word Sense Disambiguation in Gujarati Language. *International Journal of Innovative Research in Computer Science & Technology (IJIRCST), 3*(1), 1-4.

53. Dhopavkar, G., Kshirsaga, M., & Malik, L. (2014). Handling Word Sense Disambiguation in Marathi Using a Rule Based Approach. (pp. 1-4). Nagpur India: IJERA.

54. Dhopavkar, G., Kshirsaga, M., & Malik, L. (2015). Application of Rule Based Approach to Word Sense Disambiguation of Marathi Language Text. (pp. 1-5). Coimbatore, India: IEEE.

55. Dhungana, U., & Shakya, S. (2014). Word Sense Disambiguation in Nepali Language. *2014 Fourth International Conference on Digital Information and Communication Technology and its Applications (DICTAP)* (pp. 46-50). Bangkok, Thailand: IEEE.

56. Ding, Z., Yang, Z., Lin, H., & Wang, J. (2021). Focus on Interaction: A Novel Dynamic Graph Model for Joint Multiple Intent. (pp. 3801-3807). Montreal: International Joint Conferences on Artificial Intelligence Organization.

57. Domagoj, A., & Snajder, J. (2016). Cro36WSD: A Lexical Sample for Croatian Word Sense Disambiguation. *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC-2016* (pp. 1689-1694). Slovenia: European Language Resources Association (ELRA).

58. Durga Prasad, P., Sunitha, K. V., & Rani, B. P. (2018). Context-Based Word Sense Disambiguation in Telugu Using the Statistical Techniques. (pp. 271-280). Singapore: Springer.

59. Durga Prasad, P., Sunitha, K. V., & Rani, B. P. (2019). Word N-Gram Based Approach for Word Sense Disambiguation in Telugu Natural Language Processing. *International Journal of Recent Technology and Engineering (IJRTE), 7*(6S4), 1-5.

60. Elayeb, B. (2019). Arabic Word Sense Disambiguation: A Review. *Artificial Intelligence Review, 52*(4), 2475–2532.

61. Escudero, G., Màrquez, L., & Rigau, G. (2000). Boosting Applied to Word Sense Disambiguation. (pp. 129-141). Berlin, Heidelberg: Springer.

62. Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., & Smith, N. A. (2015). Retrofitting Word Vectors to Semantic Lexicons. (pp. 1606-1615). Denver, Colorado: Association for Computational Linguistics.

63. Fellbaum, C. (1998). WordNet: An Electronic Database. (pp. 1-22). Cambridge, MA: MIT Press.

64. Fellbaum, C. (2017). WordNet: An Electronic Lexical Resource. In S. Chipman (Ed.), *The Oxford Handbook of Cognitive Science* (pp. 301-311). Oxford, UK: OXFORD University Press.

65. G, H. (2004). Ontology and the Lexicon. In *Handbook on Ontologies. International Handbooks on Information Systems* (pp. 1-21). Berlin, Heidelberg: Springer.

66. Garg, S., & Mittal, A. K. (2015). A Performance of SVM with Modified Lesk Approach for Word Sense Disambiguation in Hindi Language. *International Journal of Research in Engineering and Technology, 4*(8), 1-4.

67. Garla, V. N., & Brandt, C. (2013). Knowledge-based Biomedical Word Sense Disambiguation: An Evaluation and Application to Clinical Document Classification. *Journal of the American Medical Informatics Association, 20*(5), 882–886.

68. Gautam, C. B., & Sharma, D. K. (2016). Hindi Word Sense Disambiguation Using Lesk Approach on Bigram and Trigram Words. (pp. 1-5). Bikaner, India: Association for Computing Machinery.

69. Gella, S., Strapparava, C., & Nastase, V. (2014). Mapping WordNet Domains, WordNet Topics and Wikipedia Categories to Generate Multilingual Domain Specific Resources. *In Proceedings of the 9th International Conference on Language Resources and Evaluation* (pp. 1117–1121). Reykjavik, Iceland: European Language Resources Association (ELRA).

70. Goldhahn, D., Eckart, T., & Quasthoff, U. (2012). Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages. *In:*

Proceedings of the 8th International Language Resources and Evaluation (LREC'12), 2012. Istanbul: European Language Resources Association (ELRA).

71. Gonzales, A. R., Mascarell, L., & Sennrich, R. (2017). Improving Word Sense Disambiguation in Neural Machine Translation with Sense Embeddings. *Proceedings of the Conference on Machine Translation (WMT)* (pp. 11–19). Copenhagen, Denmark: Association for Computational Linguistics.

72. Gopal, S., & Haroon, R. P. (2016). Malayalam Word Sense Disambiguation using Naïve Bayes Classifier. (pp. 1-4). Doddaballapur, India: IEEE.

73. Government, I. (2011). *CENSUS OF INDIA 2011*. New Delhi India: Office of the Registrar General & Census Commissioner, India.

74. Haque, A., & Hoque, M. (2016). Bangla Word Sense Disambiguation System using Dictionary Based Approach. (pp. 1-6). Chittagong Bangladesh: ICAICT.

75. Haroon, R. P. (2010). Malayalam Word Sense Disambiguation. (pp. 1-4). Coimbatore, India: IEEE.

76. Hassan, A. K., & Hadi, M. J. (2017). Sense-Based Information Retrieval Using Fuzzy Logic and Swarm Intelligence. *International Journal of Multimedia and Ubiquitous Engineering, 12*(1), 363-376.

77. Havasi, C., Speer, R., & Pustejovsky, J. (2010). Coarse Word-Sense Disambiguation Using Common Sense. *AAAI Fall Symposium Series 2010* (pp. 46-51). Virginia: AAAI.

78. Hristea, F., & Colhon, M. (2020). The Long Road from Performing Word Sense Disambiguation to Successfully using it in Information Retrieval: An Overview of the Unsupervised Approach. *Computational Intelligence*, 1-37.

79. Humayoun, M., & Ranta, A. (2010). Developing Punjabi Morphology, Corpus and Lexicon. *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation* (pp. 163--172). Sendai, Japan: Institute of Digital Enhancement of Cognitive Processing, Waseda University.

80. Hung, C., & Chen, S.-J. (2016). Word Sense Disambiguation based Sentiment Lexicons for Sentiment Classification. *Knowledge-Based Systems, 110*, 224-232.

81. Hutchins, J. (1999). *Warren Weaver Memorandum: 50th Anniversary of Machine Translation.* UK: MT News International.

82. Iacobacci, I., Pilehvar, M. T., & Navigli, R. (2015). Sensembed: Learning Sense Embeddings for Word and Relational Similarity. (pp. 95-105). Beijing, China: Association for Computational Linguistics.

83. Iacobacci, I., Pilehvar, M. T., & Navigli, R. (2016). Embeddings for Word Sense Disambiguation: An Evaluation Study. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (pp. 897-907). Berlin Germany: Association for Computational Linguistics.

84. Ide, N., & Véronis, J. (1998). Word Sense Disambiguation: The state of the Art. *Computational Linguistics, 24*(1), 1-40.

85. Ide, N., & Wilks, Y. (2007). Making Sense About Sense. In E. Agirre, & P. Edmonds (Eds.), *Word Sense Disambiguation Algorithms and Applications* (pp. 47-68). New York: Springer.

86. Iswarya, P., & Radha, V. (2017). Word Sense Disambiguation For Tamil Language Using Part-Of-Speech And Clustering Technique. *Journal of Engineering Science and Technology, 12*(9), 2504-2512.

87. Jain, A., & Lobiyal, D. K. (2014). A New Approach for Unsupervised Word Sense Disambiguation in Hindi Language using Graph Connectivity Measures. *International journal of artificial Intelligence and soft computing, 4*(4), 318-334.

88. Jain, A., & Lobiyal, D. K. (2015). Fuzzy Hindi WordNet and Word Sense Disambiguation using Fuzzy Graph Connectivity Measures. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 15*(2), 1-31.

89. Jain, A., & Lobiyal, D. K. (2015). Unsupervised Hindi Word Sense Disambiguation based on Network Agglomeration. (pp. 195-200). New Delhi, India: IEEE.

90. Jain, A., Yadav, S., & Tayal, D. (2013). Measuring Context-meaning for Open Class Words in Hindi Language. (pp. 118-123). Noida, India: IEEE.

91. Jain, G., & Lobiyal, D. K. (2019). Epidemic spreading based word sense disambiguation of Hindi text. Gurgaon, India: CRC Press.

92. Jain, G., & Lobiyal, D. K. (2019). Word Sense Disambiguation of Hindi Text using Fuzzified Semantic Relations and Fuzzy Hindi WordNet. Noida, India: IEEE.

93. Jimeno-Yepes, A., McInnes, B. T., & Aronson, A. R. (2011). Collocation Analysis for UMLS Knowledge-based Word Sense Disambiguation. *BMC bioinformatics, 12*(3), 1-12.

94. Jisha, P. J., Junaid, M. K., & Sherly, E. (2015). Malayalam Word Sense Disambiguation using Maximum Entropy Model. *International Journal of Engineering Research & Technology (IJERT), 3*(30), 1-4.

95. Josan, G. S., & Lehal, G. S. (2008). Size of N for Word Sense Disambiguation using N-gram Model for Punjabi Language. *International Journal of Translation, 20*(1-2), 47-56.

96. Junaid, M. K., Jisha, P. J., & Elizabeth, S. (2015). Malayalam Word Sense Disambiguation using Yamcha. (pp. 720-724). Trivandrum, India: IEEE.

97. Junaida, M. K., Jisha, P. J., & Elizabeth, S. (2017). Word Sense Disambiguation For Malayalam In A Conditional Random Field Framework. (pp. 495-502). Kolkata India: NLP Association of India.

98. Jurafsky, D., & Martin, J. H. (2009). Computational Lexical Semantics. In *Speech and Language Processing* (2nd ed., pp. 649-693). USA: Prentice-Hall, Inc.

99. Kachru, B., Kachru, Y., & Sridhar, S. (2008). *Language in South Asia*. Cambridge University Press.

100. Kakwani, D., Kunchukuttan, A., Kunchukuttan, A., N.C., G., Bhattacharyya, A., Khapra, M. M., & Kumar, P. (2020). IndicNLPSuite: Monolingual Corpora, Evaluation

Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. (pp. 4948-4961). Online: Association for Computational Linguistics.

101. Kalita, P., & Barman, A. K. (2015). Implementation of Walker Algorithm in Word Sense Disambiguation for Assamese Language. *2015 International Symposium on Advanced Computing and Communication (ISACC)* (pp. 136-140). Silchar, India: IEEE.

102. Kanojia, D., Patel, K., & Bhattacharyya, P. (2018). Indian Language Wordnets and their Linkages with Princeton WordNet. *Proceedings of the Eleventh International Conference on Language Resources and Evaluation ({LREC} 2018)* (pp. 4603-4606). Miyazaki, Japan: European Language Resources Association (ELRA).

103. Kaur, J., & Singh, A. (2015). A Customized Adaptation of Traditional Lesk Method for Sense Disambiguation of Punjabi Words from Medical Domain. *International Journal for Research in Applied Science and Engineering Technology (IJRASET), 3*(V), 1045-1051.

104. Kaur, R., Sharma, R. K., Preet, S., & Bhatia, P. (2010). Punjabi WordNet Relations and Categorization of Synsets. *In 3rd national workshop on IndoWordNet under the aegis of the 8th international conference on natural language processing (ICON 2010)* (pp. 1-6). Kharagpur, India: ICON.

105. Kaysar, M. S., Khaled, M. A., Hasan, M., & Khan, M. I. (2019). Word Sense Disambiguation of Bengali Words using FP-Growth Algorithm. *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)* (pp. 1-5). Cox'sBazar, Bangladesh: IEEE.

106. Kaysar, M., & Khan, M. (2018). Word Sense Disambiguation for Bangla Words Using Apriori Algorithm. *International Conference on Recent Advances in Mathematical and Physical Sciences(ICRAMPS)* (pp. 1-4). Dhaka, Bangladesh: PUBLONS.

107. Khapra, M. M., Bhattacharyya, P., Chauhan, S., Nair, S., & Sharma, A. (2008). Domain Specific Iterative Word Sense Disambiguation in a Multilingual Setting. (pp. 1-11). Pune India: Macmillan.

108. Khapra, M. M., Kulkarni, A., Sohoney, S., & Bhattacharyya, P. (2010). All Words Domain Adapted WSD: Finding a Middle Ground between Supervision and Unsupervision. (pp. 1-10). Uppsala, Sweden: ACL.

109. Khapra, M., Shah, S., Kedia, P., & Bhattacharyya, P. (2009). Projecting parameters for multilingual word sense disambiguation. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1* (pp. 459-467). Singapore: Association for Computational Linguistics.

110. Kilgarriff, A. (1997). I Don't Believe in Word Senses. *Comput Humanit, 31*(2), 91-113.

111. Kilgarriff, A. (2007). Word Senses. In E. Agirre, & P. Edmonds (Eds.), *Word Sense Disambiguation: Algorithms* (pp. 29–46). New York: Springer.

112. Kitchenham, B., & Stuart, C. (2007). *Guidelines for Performing Systematic Literature Reviews in Software Engineering.* UK: EBSE.

113. Koppula, N., Rani, B. P., & Rao, K. S. (2020). Word Sense Disambiguation in Telugu Language Using Knowledge-Based Approach. (pp. 153-161). Singapore: Springer.

114. Koppula, N., Rani, B., & Srinivas Rao, K. (2019). Graph-based Word Sense Disambiguation in Telugu Language. *International Journal of Knowledge-based and Intelligent Engineering Systems, 23*(1), 55-60.

115. KP, S. S., Raj, P. R., & Jayan, V. (2016). Unsupervised Approach to Word Sense Disambiguation in Malayalam. *Procedia Technology, 24*, 1507-1513.

116. Kumar, A., Mathur, I., Darbari, H., Purohit, G. N., & Joshi, N. (2016). Implications of Supervised Learning on Word Sense Disambiguation for Hindi. (pp. 1-5). Udaipur, India: ACM.

117. Kumar, B. H., M, S., & J, P. (2018). Kannada Word Sense Disambiguation by Finding the Overlaps Between the Concepts. *International Journal of Engineering & Technology, 7*(2.6), 189-192.

118. Kumar, P., & Sharma, R. K. (2012). Punjabi to UNL enconversion system. *Sadhana, 37*, 299–318.

119. Kumar, V., & Yadav, D. (2018). Word Sense-based Approach for Hindi to Tamil Machine Translation using English as Pivot Language. *Int. J. Advanced Intelligence Paradigms (IJAIP), 11*(3/4), 284-298.

120. Kumari, A., & Lobiyal, D. K. (2020). Word2vec's Distributed Word Representation for Hindi Word Sense Disambiguation. Springer.

121. Kwong, O. O. (2012). Psycholinguistics, lexicography, and word sense disambiguation. *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation* (pp. 408-417). Bali, Indonesia: Faculty of Computer Science, Universitas Indonesia.

122. Lesk, M. (1986). Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell A Pine Cone from An Ice Cream Cone. (pp. 24-26). New York, USA: SIGDOC.

123. Loureiro, D., Rezaee, K., Pilehvar, M. T., & Camacho-Collados, J. (2021). Analysis and Evaluation of Language Models for Word Sense Disambiguation. *Computational Linguistics, 47*(2), 387-443.

124. Lu, Z., Liu, T., & Li, S. (2004). Combining Neural Networks and Statistics for Chinese Word Sense Disambiguation. *Proceedings of the Third SIGHAN Workshop on Chinese Language Processing* (pp. 49-56). Barcelona, Spain: Association for Computational Linguistics.

125. Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. *Proc. icml, 30*(1), 3.

126. Manandhar, S., Klapaftis, I., Dligach, D., & Pradhan, S. (2010). SemEval-2010 Task 14: Word Sense Induction & Disambiguation. *Proceedings of the 5th International*

Workshop on Semantic Evaluation (pp. 63-68). Stroudsburg, PA, USA: Association for Computational Linguistics.

127. Manning, C. D., Raghavan, P., & Schütze, H. (2008). Scoring, term weighting and the vecetor space model. In *Introduction to Information Retrieval* (pp. 109-133). New York USA: Cambridge University Press.

128. Màrquez, L., Escudero, G., Martínez, D., & German, R. (2007). Supervised Corpus-Based Methods for WSD. In E. Agirre, & P. Edmonds (Eds.), *Word Sense Disambiguation Algorithms and Applications* (pp. 167-207). New York: Springer.

129. Martin, D. (2018). *A Semantic Unsupervised Learning Approach to Word Sense Disambiguation.* Tennessee: PhD. diss. University of Tennessee.

130. Mart'nez, D., Agirre, E., & M`arquez, L. (2002). Syntactic Features for High Precision Word Sense Disambiguation. Taipei, Taiwan: Association for Computational Linguistics.

131. Maru, M., Scozzafava, F., Martelli, F., & Navigli, R. (2019). Challenging Supervised Word Sense Disambiguation with Lexical-Semantic Combinations. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing* (pp. 3534-3540). Hong Kong, China: Association for Computational Linguistics.

132. McCarthy, D. (2006). Relating WordNet Senses for Word Sense Disambiguation. *Proceedings of the Workshop on Making Sense of Sense: Bringing Psycholinguistics and Computational Linguistics Togethe.* Association for Computing Machinery.

133. Melamud, O., Goldberger, J., & Dagan, I. (2016). Context2vec: Learning Generic Context Embedding with Bidirectional LSTM. Berlin, Germany: Association for Computational Linguistics.

134. Mihalcea, R. (2004). Co-training and Self-training for Word Sense Disambiguation. *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004* (pp. 33-40). Boston, Massachusetts, USA: Association for Computational Linguistics.

135. Mihalcea, R. (2007). Knowledge-based methods for WSD. In E. Agirre, & P. Edmonds (Eds.), *Word sense disambiguation* (pp. 107-131). New York: Springer.

136. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. New York: arXiv eprint arXiv:1301.3781{cs.CL}.

137. Miller, T., Biemann, C., Zesch, T., & Gurevych, I. (2012). Using Distributional Similarity for Lexical Expansion in Knowledge-based Word Sense Disambiguation. *Proceedings of COLING 2012* (pp. 1781-1796). Mumbai, India: The COLING 2012 Organizing Committee.

138. Mishra, N., & Siddiqui, T. J. (2012). An Investigation to Semi Supervised Approach for Hindi Word Sense Disambiguation. *Trends in Innovative Computing 2012* (pp. 126-130). Online: MIR Labs.

139. Mishra, N., Yadav, S., & Siddiqui, T. J. (2009). An Unsupervised Approach to Hindi Word Sense Disambiguation. (pp. 327-335). New Delhi India: Springer.

140. Mohammad, S., & Pedersen, T. (2004). Combining Lexical and Syntactic Features for Supervised Word Sense Disambiguation. *roceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004* (pp. 25-32). Boston, Massachusetts, USA: Association for Computational Linguistics.

141. Mohammad, S., & Pedersen, T. (2004). Complementarity of Lexical and Simple Syntactic Features: The SyntaLex Approach to SENSEVAL-3. (pp. 159-162). Barcelona, Spain: Association for Computaional Linguistics.

142. Mohan Raj, S. N., Kumar, S. S., Rajendran, S., & Soman, K. P. (2019). Word Sense Disambiguation of Malayalam Nouns. In R. Kumar, & U. K. Wiil (Eds.), *Recent Advances in Computational Intelligence* (pp. 291-314). e-book: Springer.

143. Mooney, R. (1996). Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning. *arXiv preprint cmp-lg/9612001*, 1-10.

144. Moro, A., Raganato, A., & Navigli, R. (2014). Entity Linking meets Word Sense Disambiguation: a Unified Approach. *Transactions of the Association for Computational Linguistics, 2*, 231-244.

145. Nandanwar, L. (2015). Graph Connectivity for Unsupervised Word Sense Disambiguation for HINDI Language. (pp. 1-4). Coimbatore, India: IEEE.

146. Narang, A., Sharma, R., & Kumar, P. (2013). Development of Punjabi WordNet. *CSI transactions on ICT, 1*(4), 349-354.

147. Naranjo-Zeledón, L., Ferrández A., P. J., & M, C.-R. (2019). Big Data-Assisted Word Sense Disambiguation for Sign Language. *RIIFORUM 2019: Research & Innovation Forum 2019* (pp. 441-448). Rome: Springer, Cham.

148. Narayan, D., Chakrabarti, D., Pande, P., & Bhattacharyya, P. (2002). An Experience in Building the Indo WordNet - A WordNet for Hindi. *First International Conference on Global WordNet* (pp. 1-8). Mysore, India: GWC2002.

149. Narouei, M., Ahmadi, M., & Sami, A. (2015). SePaS: Word Sense Disambiguation by Sequential Patterns in Sentences. *Natural Language Engineering, 21*(2), 251–269.

150. Naseer, A., & Hussain, S. (2009). Supervised word sense disambiguation for Urdu using Bayesian classification. *Center for Research in Urdu Language Processing, Lahore, Pakistan*, 1-5.

151. Navigli, R. (2006). Meaningful Clustering of Senses Helps Boost Word Sense Disambiguation Performance. *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics* (pp. 105-112). Sydney, Australia: Association for Computational Linguistics.

152. Navigli, R. (2009). Word Sense Disambiguation: A Survey. *ACM Comput. Surv, 41*(ACM), 1-69.

153. Navigli, R., & Ponzetto, S. P. (2012). BabelNet: The Automatic Construction, Evaluation and Application of a Wide-coverage Multilingual Semantic Network. *Artificial Intelligence, 193*, 217-250.

154. Navigli, R., & Velardi, P. (2005). Structural Semantic Interconnections: A Knowledge-based Approach to Word Sense Disambiguation. (pp. 1075-1086). USA: IEEE Computer Society.

155. Navigli, R., Bevilacqua, M., Conia, S., Montagnini, D., & Cecconi, F. (2021). Ten Years of BabelNet: A Survey. (pp. 4559-4567). Montreal: International Joint Conferences on Artificial Intelligence Organization.

156. Nazah, S., Hoque, M., & Hossa. (2017). Word sense disambiguation of Bangla sentences using statistical approach. *2017 3rd International Conference on Electrical Information and Communication Technology* (pp. 1-6). Khulna, Bangladesh: IEEE.

157. Niles, I., & Pease, A. (2001). Towards a Standard Upper Ontology. (pp. 2-9). Ogunquit, Maine, USA: Association for Computing Machinery.

158. Nithyanandan, S., & C, R. (2019). Deep Learning Models For Word Sense Disambiguation: A Comparative Study. *In proceedings of the International Conference on Systems, Energy & Environment (ICSEE) 2019* (pp. 1-5). Kannur, Kerala: SSRN.

159. Oele, D., & Van Noord, G. (2017). Distributional Lesk: Effective Knowledge-based Word Sense Disambiguation. *IWCS 2017—12th International Conference on Computational Semantics—Short papers* (pp. 1-8). IWCS: ACL Anthology.

160. Padmamala, R. (2015). Word Level Translation (Tamil - English) With Word Sense Disambiguation in Tamil using OntNet. (pp. 191-198). Chennai, India: IEEE.

161. Pal, A. R., & Saha, D. (2016). Word Sense Disambiguation in Bengali: An Auto-updated Learning Set Increases the Accuracy of the Result. *Information Systems Design and Intelligent Applications* (pp. 423-430). New Delhi: Springer India.

162. Pal, A. R., & Saha, D. (2017). Word Sense Disambiguation in Bengali: An Unsupervised Approach. *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)* (pp. 1-5). Coimbatore: IEEE.

163. Pal, A. R., Saha, D., & Dash, N. S. (2015). Automatic Classification of Bengali Sentences Based on Sense Definitions Present in Bengali Wordnet. *International Journal of Control Theory and Computer Modeling, 5*(1), 1-13.

164. Pal, A. R., Saha, D., & Pal, A. (2017). A Knowledge based Methodology for Word Sense Disambiguation for Low Resource Language. *Advances in Computational Sciences and Technology, 10*(2), 267-283.

165. Pal, A. R., Saha, D., Dash, N. S., Naskar, S. K., & Pal, A. (2019). A Novel Approach to Word Sense Disambiguation in Bengali Language using Supervised Methodology. *Sādhanā, 44*(181), 1-12.

166. Pal, A. R., Saha, D., Naskar, S., & Dash, N. S. (2015). Word Sense Disambiguation in Bengali: A Lemmatized System Increases the Accuracy of the Result. (pp. 342-346). Kolkata: IEEE.

167. Pal, A., & Saha, D. (2019). Word Sense Disambiguation in Bengali Language Using Unsupervised Methodology With Modifications. *Sādhanā, 44*(7), 168.

168. Palanati, D., & Kolikipogu, R. (2013). Decision List Algorithm for Word Sense Disambiguation for TELUGU Natural Language Processing. *Int. J. Electron. Commun. Comput. Eng*, *4*(6), 176-180.

169. Palmer, M., Ng, H. T., & Dang, H. T. (2007). Making Fine-grained and Coarse-grained Sense Distinctions Both Manually and Automatically. *Natural Language Engineering, 13*(2), 137-163.

170. Pamungkas, E. W., & Putri, D. G. (2017). Word Sense Disambiguation for Lexicon-Based Sentiment Analysis. *Proceedings of the 9th International Conference on Machine Learning and Computing* (pp. 442–446). Singapore: Association for Computing Machinery.

171. Pandit, R., & Naskar, S. K. (2015). A Memory Based Approach to Word Sense Disambiguation in Bengali Using k-NN Method. (pp. 383-386). Kolkata: 2015 IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS).

172. Parameswarappa, S., & Narayana, V. (2011). Kannada Word Sense Disambiguation for Machine Translation. *International Journal of Computer Applications, 34*(10), 1-8.

173. Parameswarappa, S., & Narayana, V. N. (2011). Kannada Word Sense Disambiguation Using Association Rules. (pp. 47-56). Heidelberg Berlin: Springer.

174. Parameswarappa, S., & Narayana, V. N. (2011). Target word sense disambiguation system for Kannada language. Bangalore, India: IET.

175. Parameswarappa, S., & Narayana, V. N. (2012). Kannada Verb and their Automatic Sense Disambiguation. (pp. 1-8). Mastue Japan: Globle WordNet Conference.

176. Parameswarappa, S., Narayana, V., & Yarowsky, D. (2013). Kannada Word Sense Disambiguation Using Decision List. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 2*(3), 272-278.

177. Patel, N., Patel, B., Parikh, R., & Bhatt, B. (2015). Hierarchical Clustering Technique for Word Sense Disambiguation Using Hindi WordNet. (pp. 1-5). Ahmedabad, India: IEEE.

178. Patil, A., Patil, C., Ramteke, R., R P, B., & Darbari, H. (2020). Exploring Resources in Word Sense Disambiguation for Marathi Language. *International Research Journal on Advanced Science Hub, 2*, 108-111.

179. Patwardhan, S., & Pedersen, T. (2006). Using WordNet-based Context Vectors to Estimate the Semantic Relatedness of Concepts. *Proceedings of the Workshop on Making Sense of Sense: Bringing Psycholinguistics and Computational Linguistics Together* (pp. 1-8). Trento, Italy: Association for Computational Linguistics.

180. Pedersen, t. (2007). Unsupervised Corpus-Based Methods for WSD. In E. Agirre, & P. Edmonds (Eds.), *Word Sense Disambiguation Algorithms and Applications* (pp. 133-162). New York: Springer.

181. Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). WordNet::Similarity: Measuring the Relatedness of Concepts. *HLT-NAACL--Demonstrations '04* (pp. 38--41). Boston, Massachusetts: Association for Computational Linguistics.

182. Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 conference on empirical methods in natural language processing* (pp. 1532-1543). Doha, Qatar: Association for Computational Linguistics.

183. Pianta, E., Bentivogli, L., & Girardi, C. (2002). MultiWordNet Developing an aligned multilingual database. Mysore, India: International WordNet Conference.

184. Ponzetto, S., & Navigli, R. (2010). Knowledge-rich Word Sense Disambiguation Rivaling Supervised Systems. *Proceedings of the 48th annual meeting of the association for computational linguistics* (pp. 1522-1531). Uppsala, Sweden: Association for Computational Linguistics.

185. Popov, A. (2018). Neural Network Models for Word Sense Disambiguation: An Overview. *Cybernetics and Information Technologies, 18*(1), 139-151.

186. Preiss, J., Dehdari, J., King, J., & Mehay, D. (2009). Refining the most frequent sense baseline. *Proceedings of the NAACL HLT Workshop on Semantic Evaluations: Recent Achievements and Future Directions* (pp. 10-18). Boulder, Colorado: Association for Computational Linguistics.

187. Pu, X., Pappas, N., & Popescu-Belis, A. (2017). Sense-Aware Statistical Machine Translation using Adaptive Context-Dependent Clustering. *Proceedings of the Conference on Machine Translation (WMT)* (pp. 1-10). Copenhagen, Denmark: Association for Computational Linguistics.

188. Purandare, A., & Pedersen, T. (2004). Word Sense Discrimination by Clustering Contexts in Vector and Similarity Spaces. (pp. 41-48). Boston: Association for Computational Linguistics.

189. Quinlan, J. R. (1996). Bagging Boosting and C4.5. California USA: AAAI.

190. Raganato, A., Delli Bovi, C., & Navigli, R. (2017). Neural Sequence Learning Models for Word Sense Disambiguation. (pp. 1156-1167). Copenhagen, Denmark: Association for Computational Linguistics.

191. Rahman, N., & Borah, B. (2020). Improvement of Query-Based Text Summarization Using Word Sense Disambiguation. *Complex & Intelligent Systems, 6*, 75-85. doi:https://doi.org/10.1007/s40747-019-0115-2

192. Rais, M., & Lachkar, A. (2018). An Empirical Study of Word Sense Disambiguation for Biomedical Information Retrieval System. *Bioinformatics and Biomedical Engineering* (pp. 314-326). Granada, Spain: Springer.

193. Rajendran, S., Anand Kumar, M., & K P, S. (2013). Computational Approach to Word Sense Disambiguation in Tamil. *12th International Tamil Internet Conference 2013* (pp. 35-44). Kuala Lumpur, Malaysia: INFITT.

194. Rana, P., & Kumar, P. (2015). Word Sense Disambiguation for Punjabi Language Using Overlap Based Approach. In *Advances in Intelligent Informatics* (pp. 607-619). Springer.

195. Rao, R., & Kallimani, J. (2016). Analysis of Polysemy Words in Kannada Sentences Based on Parts of Speech. (pp. 500-504). Jaipur, India: IEEE.

196. Rastogi, P., & Dwivedi, S. K. (2011). Performance Comparison of Word Sense Disambiguation (wsd) Algorithm on Hindi Language Supporting Search Engines. *International Journal of Computer Science Issues (IJCSI), 8*(2), 375-379.

197. Rehurek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. 3*, pp. 45-50. Valletta, Malta: ELRA.

198. Roget, P. M. (1911). *Roget's International Thesaurus* (1st ed.). New York: Cromwell.

199. Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review, 33*, 1–39.

200. Ron, A., & Poesio, M. (2008). Inter-coder Agreement for Computational Linguistics. *Computational Linguistics, 34*(4), 555–596.

201. Rothe, S., & Schütze, H. (2015). Autoextend: Extending Word Embeddings to Embeddings for Synsets and Lexemes. (pp. 1793-1803). Beijing, China: Association for Computational Linguistics.

202. Roy, A., Sarkar, S., & Purkayastha, B. (2014). Knowledge Based Approaches To Nepali Word Sense Disambiguation. *International Journal on Natural Language Computing, 3*, 51-63. doi:10.5121/ijnlc.2014.3305

203. Rumelhart, D. E., & McClelland, J. L. (1987). *Learning Internal Representations by Error Propagation.* Cambridge, Massachusetts: Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations , MITP.

204. Saeed, A., Nawab, R. M., Stevenson, M., & Rayson, P. (2019). A Sense Annotated Corpus for All-Words Urdu Word Sense Disambiguation. *ACM Trans. Asian Low-Resour. Lang. Inf. Process., 18*(4), 1-14.

205. Saeed, A., Nawab, R. M., Stevenson, M., & Rayson, P. (2019). A Word Sense Disambiguation Corpus for Urdu. *Lang Resources & Evaluation, 53*, 397-418. doi:https://doi.org/10.1007/s10579-018-9438-7

206. Sankaran, B., & Vijay-Shanker, K. (2003). Influence of Morphology in Word Sense Disambiguation for Tamil.

207. Sarika, & Sharma, D. K. (2016). Hindi Word Sense Disambiguation Using Cosine Similarity. (pp. 801-808). Singapore: Springer.

208. Sarmah, J., & Sarma, S. (2016). Decision tree based supervised word sense disambiguation for Assamese. *Int. J. Comput. Appl, 141*(1), 42-48.

209. Sarmah, J., & Sarma, S. K. (2016). Word Sense Disambiguation for Assamese. *2016 IEEE 6th International Conference on Advanced Computing (IACC)* (pp. 146-151). Bhimavaram, India: IEEE.

210. Sarmah, J., & Sarma, S. K. (2018). *Word Sense Disambiguation for Assamese Language.* Gauhati University. http://hdl.handle.net/10603/246293: Shodhganga.

211. Sau, A., Amin, T., Barman, N., & Pal, A. (2019). Word Sense Disambiguation in Bengali Using Sense Induction. *2019 International Conference on Applied Machine Learning* (pp. 170-174). Bhubaneswar, India: IEEE.

212. Sawhney, R., & Kaur, A. (2014). A Modified Technique for Word Sense Disambiguation Using Lesk Algorithm in Hindi Language. (pp. 2745-2749). New Delhi, India: IEEE.

213. Schuemie, M., Kors, J., & Mons, B. (2005). Word Sense Disambiguation in the Biomedical Domain: An Overview. *J Comput Biol, 12*(5), 554-65.

214. Schwab, D., Tchechmedjiev, A., Goulian, J., Nasiruddin, M., S'erasset, G., & Blanchon, H. (2013). GETALP System: Propagation of A Lesk Measure Through An Ant Colony Algorithm. *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)* (pp. 232-240). Atlanta, Georgia, USA: Association for Computational Linguistics.

215. Seifollahi, S., & Shajari, M. (2019). Word Sense Disambiguation Application in Sentiment Analysis of News Headlines: An Applied Approach to FOREX Market Prediction. *Journal of Intelligent Information Systems, 52*(1), 57–83.

216. Sengupta, S., Pandit, R., Mitra, P., Naskar, S., & Sardar, M. (2019). Word Sense Induction in Bengali Using Parallel Corpora and Distributional Semantics. *Journal of Intelligent & Fuzzy Systems, 36*(5), 4821-4832.

217. Sesmero, M. P., Ledezma, A. I., & Sanchis, A. (2015). Generating Ensembles of Heterogeneous Classifiers Using Stacked Generalization. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 5*(1), 21-34.

218. Sharma, P., & Joshi, N. (2019). Knowledge-Based Method for Word Sense Disambiguation by Using Hindi WordNet. *Engineering, Technology & Applied Science Research, 9*(2), 3985-3989.

219. Shashank, N., & Kallimani, J. (2017). Word Sense Disambiguation of Polysemy Words in Kannada Language. (pp. 641-644). Udupi, India: IEEE.

220. Shrestha, A., & Mahmood, A. (2019). Review of Deep Learning Algorithms and Architectures. *IEEE Access, 7*, 53040-53065.

221. Shrestha, N., Hall, P., & Bista, S. (2008). Resources for Nepali Word Sense Disambiguation. *2008 International Conference on Natural Language Processing and*

Knowledge Engineering (pp. 1-5). Beijing, China: IEEE. doi:10.1109/NLPKE.2008.4906758

222. Singh, J., & Singh, I. (2015). Word Sense Disambiguation: Enhanced Lesk Approach in Punjabi Language. *International Journal of Computer Applications, 129*(6), 23-27.

223. Singh, R., Ghosh, K., Nongmeikapam, K., & Bandyopadhyay, S. (2014). A Decision Tree Based Word Sense Disambiguation System in Manipuri Language. *Advanced Computing, 5*(4), 1-17.

224. Singh, S., & Siddiqui, T. (2015). Utilizing Corpus Statistics for Hindi Word Sense Disambiguation. *Int. Arab J. Inf. Technol., 12*(6A), 755-763.

225. Singh, S., & Siddiqui, T. J. (2012). Evaluating Effect of Context Window Size, Stemming and Stop Word Removal on Hindi Word Sense Disambiguation. (pp. 1-5). Kuala lumpur Malaysia: IEEE.

226. Singh, S., & Siddiqui, T. J. (2015). Role of a Semantic Relations in Hindi Word Sense Disambiguation. *Procedia Computer Science, 46*, 240-248.

227. Singh, S., & Siddiqui, T. J. (2015). Role of Karaka Relations in Hindi Word Sense Disambiguation. *Journal of Information Technology Research (JITR), 8*(3), 21-42.

228. Singh, S., Siddiqui, T. J., & Sharma, S. K. (2014). Naive Bayes Classifier for Hindi Word Sense Disambiguation. (pp. 1-8). Nagpur, India: Association for Computing Machinery.

229. Singh, S., Singh, V. K., & Siddiqui, T. J. (2013). Hindi Word Sense Disambiguation Using Semantic Relatedness Measure. (pp. 247-256). Berlin, Heidelberg: Springer.

230. Singh, U. P., Goyal, V., & Rani, A. (2014). Disambiguating Hindi Words Using N-Gram Smoothing Models. *International Journal of Engineering Sciences, 10*, 26-29.

231. Singh, V. p., & Kumar, P. (2018). Naive Bayes Classifier for Word Sense Disambiguation of Punjabi Language. *Malaysian Journal of Computer Science, 31*(3), 188–199.

232. Singh, V. p., & Kumar, P. (2019). Word Sense Disambiguation for Punjabi Language Using Deep Learning Techniques. *Neural Computing and Applications*, 1-11. doi: https://doi.org/10.1007/s00521-019-04581-3

233. Singh, V., & Kumar, P. (2019). Sense Disambiguation for Punjabi Language Using Supervised Machine Learning Techniques. *Sādhanā, 44*(226), 1-15.

234. Sinha, M., Reddy, M. K., Bhattacharya, P., Pandey, P., & Kashyap, L. (2004). Hindi Word Sense Disambiguation. *In Proceedings of the International Symposium on Machine Translation, Natural Language Processing, and Translation Support Systems* (pp. 1-7). Delhi: IIT Bombay.

235. Snow, R., Prakash, S., Jurafsky, D., & Ng, A. Y. (2007). Learning to Merge Word Senses. *In Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning* (pp. 1005-1014). Prague, Czech Republic: Association for Computational Linguistics.

236. Soanes, C., & Stevenson, A. (2003). *Oxford Dictionary of English.* Oxford, UK: Oxford University Press.

237. Sreedhar, J., Viswanadha Raju, S., & Vinaya Babu , A. (2014). Two-Word and Three-Word Disambiguation Rules for Telugu Language Sentences: A Practical Approach. *Global Journal of Comuter Science and Technology: G Interdisciplinary, 14*(1), 1-7.

238. Su`arez, A., & Palomar, M. (2002). A Maximum Entropy-Based Word Sense Disambiguation System. (pp. 1-7). Taipei, Taiwan: Association for Computational Linguistics.

239. Taghipour, K., & Ng, H. (2015). Semi-supervised word sense disambiguation using word embeddings in general and specific domains. *Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: human language technologies* (pp. 314-323). Denver, Colorado: Association for Computational Linguistics.

240. Tayal, D. K., Ahuja, L., & Chhabra, S. (2015). Word Sense Disambiguation in Hindi Language Using Hyperspace Analogue to Language and Fuzzy C-Means Clustering. (pp. 49-58). Trivandrum, India: NLP Association of India.

241. Tratz, S., Sanfilippo, A., Gregory, M., Chappell, A., Posse, C., & Whitney, P. (2007). PNNL: A Supervised Maximum Entropy Approach to Word Sense Disambiguation. (pp. 264-267). Prague: Association for Computational Linguistics.

242. Tripathi, P., Mukherjee, P., Hendre, M., Godse, M., & Chakraborty, B. (2020). Word Sense Disambiguation in Hindi Language Using Score Based Modified Lesk Algorithm. *International Journal of Computing and Digital Systems, 10*, 2-20.

243. Tufis, D., Cristea, D., & Stamou. (2004). BalkaNet: Aims, Methods,Results and Perspectives.A General Overview. *Romanian Journal Of Information Science And Technology, 7*(1-2), 9-43.

244. Unnikrishnan, N., & P, B. (2002). Word Sense Disambiguation Using Semantic Graphs. *International Conference on Global WordNet.* Mysore.

245. Vaishnav, Z. (2017). Gujarati Word Sense Disambiguation using Genetic Algorithm. *International Journal on Recent and Innovation Trends in Computing and Communication, 5*(6), 635-639.

246. Vaishnav, Z., & Sajja , P. (2019). Knowledge-Based Approach for Word Sense Disambiguation Using Genetic Algorithm for Gujarati. *Information and Communication Technology for Intelligent Systems. Smart Innovation, Systems and Technologies. 106*, pp. 485-494. Singapore: Springer.

247. Van de Cruys, T., & Apidianaki, M. (2011). Latent Semantic Word Sense Induction and Disambiguation. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1* (pp. 1476-1485). Stroudsburg, PA, USA: Association for Computational Linguistics.

248. Vasilescu, F., Philippe, L., & Guy, L. (2004). Evaluating Variants of the Lesk Approach for Disambiguating Words. (pp. 633-636). Lisbon, Portugal: European Language Resources Association (ELRA).

249. Veronis, J. (2004). Hyperlex: Lexical Cartography for Information Retrieval. *Comput. Speech & Language, 18*(3), 223-252.

250. Véronis, J., & Ide, N. M. (1990). Word Sense Disambiguation with Very Large Neural Networks Extracted from Machine Readable Dictionaries. *Proceedings of the 13th conference on Computational linguistics-Volume 2* (pp. 389-394). Helsinki, Finland: Association for Computational Linguistics.

251. Vial, L., Lecouteux, B., & Schwab, D. (2019). Sense Vocabulary Compression through the Semantic Knowledge of {W}ord{N}et for Neural Word Sense Disambiguation",. (pp. 108-117). Wroclaw, Poland: Global Wordnet Association.

252. Vintar, Š., & Fišer, D. (2016). Using WordNet-Based Word Sense Disambiguation to Improve MT Performance. In M. Costa-jussà, R. Rapp, P. Lambert, K. Eberle, R. Banchs, B. Babych, M. Costa-jussà, R. Rapp, P. Lambert, K. Eberle, R. Banchs, & B. Babych (Eds.), *Hybrid Approaches to Machine Translation. Theory and Applications of Natural Language Processing* (pp. 191-205). Switzerland: Springer, Cham.

253. Virk, S. M., Humayoun, M., & Ranta, A. (2011). An Open Source Punjabi Resource Grammar. *Proceedings of Recent Advances in Natural Language Processing* (pp. 70-76). Hissar, Bulgaria: ACM.

254. Vossen, P. (2002). Wordnet, eurowordnet and global wordnet. *Revue fran {\ c {c}} comfortable with applied linguistics {\ 'e} e, 7*(1), 27--38.

255. Vossen, P., Bond, F., & McCrae, J. (2016). Toward a truly multilingual Global Wordnet Grid. Bucharest, Romania: Global WordNet Conference.

256. Vyas, T., & Ganatra, A. (2019). Gujarati Language: Research Issues, Resources and Proposed Method on Word Sense Disambiguation. *International Journal of Recent Technology and Engineering (IJRTE), 8*(2S11), 1-5.

257. Walia, H., Rana, A., & Kansal, V. (2017). A Naïve Bayes Approach for working on Gurmukhi Word Sense Disambiguation. (pp. 432-435). Noida, India: IEEE.

258. Walia, H., Rana, A., & Kansal, V. (2018). A Supervised Approach on Gurmukhi Word Sense Disambiguation Using K-NN Method. (pp. 743-746). Noida, India: IEEE.

259. Walia, H., Rana, A., & Kansal, V. (2018). Word Sense Disambiguation: Supervised Program Interpretation Methodology for Punjabi Language. (pp. 762-767). Noida, India: IEEE.

260. Walia, H., Rana, A., & Kansal, V. (2019). Case Based Interpretation Model for Word Sense Disambiguation in Gurmukhi. (pp. 359-364). Noida, India: IEEE.

261. Wang, T., & Hirst, G. (2014). Applying a Naive Bayes Similarity Measure to Word Sense Disambiguation. *Proceedings of the 52nd Annual Meeting of the Association for*

Computational Linguistics (Volume 2: Short Papers) (pp. 531-537). Baltimore, Maryland, USA: Association for Computational Linguistics.

262. Witten, I. H., Frank, E., & Hall, M. A. (2011). Implementations: Real Machine Learning Schemes. In I. H. Witten, E. Frank, & M. A. Hall, *Data Mining Practical Machine Learning Tools and Techniques* (pp. 191-304). Burlington, MA: Morgan Kaufmann.

263. Xiu, Y., Lan, M., & Wu, Y. (2017). Using Supervised and Unsupervised Methods to Detect and Locate English Puns. (pp. 453-456). Vancouver, Canada: Association for Computational Linguistics.

264. Yadav, P., & Vishwakarma, S. (2013). Mining Association Rules Based Approach to Word SenseDisambiguation for Hindi Language. *International Journal of Emerging Technology and Advanced Engineering, 3*(5), 470-473.

265. Yarowsky, D. (2000). Hierarchical Decision Lists for Word Sense Disambiguation. *Comput. Human, 34*(1-2), 179-186.

266. Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *IEEE Computational intelligence magazine, 13*(3), 55--75.

267. Yuan, D., Richardson, J., Doherty, R., Evans, C., & Altendorf, E. (2016). Semi-supervised word sense disambiguation with neural models. *arXiv preprint arXiv:1603.07012*, 1-12.

268. Zhong, Z., & Ng, H. (2010). It makes sense: A wide-coverage word sense disambiguation system for free text. *Proceedings of the ACL 2010 system demonstrations* (pp. 78-83). Uppsala, Sweden: Association for Computational Linguistics.

269. Zhou, Z.-H. (2012). *Ensemble methods: foundations and algorithms.* New York, London: CRC press.

270. Zungre, N., & Dhopavkar, G. (2016). Sense disambiguation for Marathi language words using decision graph method. Coimbatore, India: IEEE.

271. Zuppann, A. (2003). A Connectionist Approach to Word Sense Disambiguation. *In Class of 2003 Senior Conference on Natural Language Processing.* Swarthmore Pennsylvania, USA: CSE Swarthmore College .