Journal of Trends in Computer Science and Smart Technology (ISSN: 2582-4104)
www.irojournals.com/tcsst/

Evaluating Random Forest and Decision Tree Algorithms for Resolving Lexical Ambiguity in the Gujarati Language

Avani N Dave[1], Sanjay M Shah[2], Nakul R Dave[3]

[1]Research Scholar, Computer/IT Engineering, Gujarat Technological University, Ahmedabad, India.
[2]Professor, Computer Engineering Department, Government Engineering College, Rajkot, India.
[3]Assistant Professor, Computer Engineering Department, Vishwakarma Government Engineering College, Ahmedabad, India.

Email: [1]avanindave@gmail.com, [2]sanjay_shah_r@yahoo.com, [3]davenakulr@gmail.com

Orcid ID: [1]0000-0003-2021-2302, [2]0000-0002-4937-4527, [3]0000-0003-1124-912X

Abstract

Word sense disambiguation is the task of determining the exact meaning of a word based on its context. This task is crucial in natural language processing. The lack of labeled datasets and the complex structure of the language, which includes idiomatic usage and subtle semantic changes, contribute to the poor outcomes of earlier attempts to solve word sense disambiguation in Gujarati. As a result, various models have shown low accuracy. To address this issue, we have created a new dataset that is manually sense-annotated for unclear Gujarati words. The corpus contains 50 ambiguous words, and each word has been assigned to the appropriate context. This makes it a valuable starting point for evaluating supervised learning models. With this newly compiled corpus, we carry out a systematic study of two supervised machine learning algorithms-Decision Tree and Random Forest-using 3-fold and 5-fold cross-validation. Our results show that Random Forest obtains the highest accuracy, highlighting which supervised methods are best suited for this particular task. The main contributions of this work include the development of a much-needed annotated corpus and sufficient evidence to prove that supervised learning can be quite effective in improving WSD for Gujarati when proper data is integrated.

Keywords: Machine Learning, Word Sense Disambiguation, Natural Language Processing, Decision Tree, Random Forest, Sense Annotated Corpus, Gujarati Language.

1. Introduction

A key area of research in artificial intelligence (AI) is natural language processing (NLP), which allows computers to comprehend, interpret, and produce human language [1-3]. Word sense disambiguation (WSD), which involves determining a word's proper meaning (or "sense") based on its context, is one of the oldest and most challenging NLP problems.

The ability to handle lexical ambiguity is a critical component in a wide range of sophisticated applications of NLP such as machine translation, information retrieval, sentiment analysis, and question-answering systems. For instance, in the sentence "He sat on the river bank," the WSD system should identify that bank refers to the land along or beside a river, not

DOI: https://doi.org/10.36548/jtcsst.2025.4.006
Received: 03.10.2025, received in revised form: 04.11.2025, accepted: 19.11.2025, published: 02.12.2025

a financial institution. Although significant strides have been achieved for high-resource languages like English, this is mainly due to the availability of large-scale, sense-tagged corpora such as SemCor. In the case of low-resource languages, the challenge remains unsolved [4]. Gujarati, spoken by over 55 million people in the world, is a morphologically rich and low-resource language where WSD research is in its infancy [5], Gujarati has a complex nature typified by subtlety in semantic distinctions and the frequent occurrence of idiomatic expressions; this enhances the difficulties associated with the WSD task. Table 1 gives an example of a classic ambiguity within the Gujarati language.

Table 1. Example of Lexical Ambiguity for the Gujarati Word "હાર" (Haar)

Meaning	Gujarati Sentence	Transliteration	English Translation
Defeat	મેચમાં એક ટીમની હાર થઈ.	Mēcamāṁ ēka ṭīmanī hār thaī.	One team was defeated in the match.
Necklace	રાણીએ ગળામાં હીરાનો હાર પહેર્યો હતો.	Rāṇī'ē gaḷāmāṁ hīrānō hār pahēryō hatō.	The queen wore a diamond necklace.

1.1 Motivation

The primary motivation for this research lies in the fact that there is a critical bottleneck that makes improvement in Gujarati NLP difficult. While data-driven, supervised machine learning approaches have now emerged as the de facto standard for state-of-the-art performance in WSD for high-resource languages like English, it remains a significant challenge to apply these approaches to low-resource languages [2]. Despite its large speaker base, Gujarati lacks the fundamental computational resources that have catalyzed progress in other languages. This has motivated efforts to build WSD systems for several other Indian languages, such as Hindi and Bengali, driven primarily by the creation of dedicated WordNets and annotated corpora [6]. These resources have finally allowed researchers to go beyond purely knowledge-based methods and consider the more powerful supervised and neural paradigms.

While there is a publicly available "gold-standard" corpus for WSD in other languages, the Gujarati NLP ecosystem has none. Due to this fact, studies on Gujarati WSD have been limited to knowledge-based or unsupervised approaches that, though instructive when data is scarce, inherently lag behind supervised models on complex contextual patterns learned by the latter. Also, as there is no "gold-standard" sense-annotated corpus in Gujarati,

- There is a limitation in effectively training or evaluating supervised models, often the most powerful.
- There is no standard benchmark against which different WSD algorithms can be compared, which complicates work on assessing the progress that is being made.
- Higher-level applications, like accurate machine translation and sophisticated chatbots for Gujarati, have yet to be developed.

The motivation for this work is the pressing need for this foundational resource to create a strong performance baseline for supervised WSD in Gujarati, so that further research and development can be facilitated.

1.2 Research Questions

The core research questions that this study sought to address include:

1. Can a manually sense-annotated corpus for ambiguous Gujarati words be effectively developed to act as a reliable benchmark for supervised WSD tasks?
2. How well do the standard supervised machine learning algorithms, namely Decision Tree (DT) and Random Forest (RF), perform on this new corpus for the task of Gujarati WSD?
3. Among the models evaluated, which algorithm performs the best, and what does its performance indicate about the most suitable computational approach that can be used to resolve lexical ambiguity in a morphologically rich language like Gujarati?

1.3 Contributions

The key contributions of this paper are threefold and visualized in Figure 1:

1. A Novel Sense-Annotated Corpus: For the first time, we create a manually sense-annotated corpus for 50 ambiguous Gujarati words, named Guj-Sense. This resource is an essential benchmark for training and evaluating supervised WSD models, filling a very significant resource gap in Gujarati NLP.
2. Extensive Empirical Study: We carry out a detailed comparative analysis of two popular supervised learning algorithms applied to the Guj-Sense corpus. To the best of our knowledge, this is the first systematic study that assesses and compares the effectiveness of these varied classifiers for the Gujarati WSD task on a standardized dataset.
3. A Performance Benchmark and Key Insights: Our results set a very strong performance benchmark for future research. We show that the RF algorithm outperforms other models by a great margin, therefore providing valuable insights into how ensemble methods perform in capturing the nuances of the Gujarati language.

1.4 Paper Structure

The remainder of this paper is organized as follows. Section 2 provides a review of related work in Word Sense Disambiguation, focusing on both global and Indian language contexts. Section 3 details an overview of the WSD for the Gujarati language, considering various supervised algorithms. Section 4 outlines a model training and testing framework for supervised Gujarati WSD. Section 5 presents the design and development of a sense-annotated corpus, the experimental results, the evaluation metrics, and the analysis. Finally, Section 6 concludes with key findings and directions for future research.

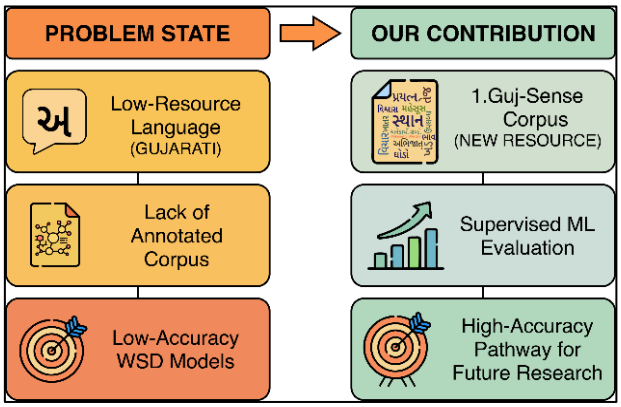


Figure 1. Contribution of the Research Paper

2. Related Work

WSD is a crucial task in NLP, aiming to clarify lexical ambiguity by determining the appropriate meaning of a word based on context. The difficulty of this task varies significantly across languages, due to differences in intrinsic linguistic frameworks and the availability of computational resources. Indian languages, such as Gujarati and Hindi, pose significant challenges for word-sense disambiguation due to their complex morphology, syntactic variability, and frequent use of compound expressions. A considerable aspect exacerbating this challenge is the lack of extensive lexical resources and annotated corpora. In sharp contrast, although WSD is a barrier for higher-resource languages, their advancement has been significantly expedited by robust resources such as WordNet and comprehensive sense-tagged datasets, which are predominantly underdeveloped or inaccessible for their Indian equivalents.

WSD is a computational methodology designed to identify the precise meaning, or "sense," of a polysemous word within a specific context [7]. The dual objective of WSD is to first determine the full inventory of possible senses for a word and then accurately assign the contextually appropriate sense, a crucial step in enhancing a machine's semantic comprehension. A conceptual WSD framework typically integrates several key knowledge sources to resolve ambiguity. It leverages structured lexical resources, such as WordNet, for sense definitions, analyses contextual knowledge from surrounding text and grammatical structures, and often employs machine learning algorithms to discern disambiguation patterns from the data. By synthesising these diverse inputs, the system performs a final matching process to assign the most plausible meaning to the ambiguous word, as shown in Figure 2 [8].

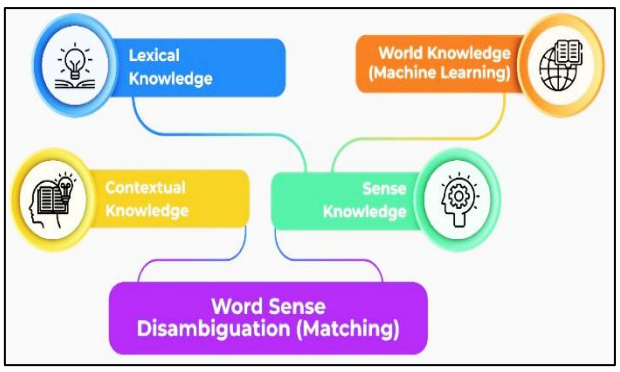


Figure 2. Conceptual Model of Word Sense Disambiguation

To approach WSD effectively, a thorough understanding of three core concepts is essential: polysemy and homonymy, contextual knowledge, and sense knowledge. Polysemy refers to words that possess multiple, related meanings, while homonymy describes words that have the exact spelling but different, unrelated meanings. For instance, the Gujarati word "જગ" (Jag) can signify either "વિશ્વ" (Vishva, world) or "પાણીનું પાત્ર" (Pāṇīnuṁ pātra, jug or pitcher), depending on its context of use. Consider these illustrative examples: Phrase 1: "This world is wonderful." (આ જગ ખુબ સુંદર છે. / Ā jag khuba sundara chē.) Phrase 2: "Fill this jug with water." (આ જગમાં પાણી ભરો. / Ā jagmāṁ pāṇī bharo.) In the first sentence, "જગ" clearly refers to "વિશ્વ" ("world"), whereas in the second, the same word denotes a container or "પાણીનું પાત્ર" ("jug"). This ambiguity necessitates that WSD accurately identify the correct meaning given the context. Contextual knowledge plays a crucial role in resolving ambiguity by considering the surrounding words, sentence structure, and broader discourse context. Finally, sense knowledge encompasses the detailed inventory of all possible interpretations or meanings associated with a particular word. Figure 3 represents a word cloud highlighting the most frequently encountered polysemous words in Gujarati [1]. To systematically address sense ambiguity, lexical databases such as WordNet have been widely utilized. Gujarati WordNet, for instance, systematically arranges words into synonym groups called "synsets," with each synset representing a distinct concept or gloss, thus offering an organized and comprehensive sense inventory for the language.

Figure 3. Word Cloud of અનેકાર્થી શબ્દો (Anēkārthī śabdō / Polysemy Words) for the Gujarati Language

Research in WSD has advanced along several distinct methodological paradigms, which can be broadly classified as knowledge-based, supervised, unsupervised, and hybrid approaches [10, 11]. These methods differ fundamentally in their reliance on annotated data and external lexical resources [12–13]. Supervised methods utilize manually labeled corpora to train predictive models, whereas unsupervised techniques infer word senses from unannotated text, often in conjunction with resources such as WordNet [15–24]. Semi-supervised approaches bridge this gap by combining labeled and unlabeled data, which is particularly effective in resource-constrained scenarios [23–25]. Unsupervised WSD utilizes techniques that are independent of annotated data. These strategies employ unannotated text and external knowledge sources to ascertain the precise meanings of words [26–29].

Furthermore, knowledge-based systems depend exclusively on lexical resources, and hybrid models integrate multiple strategies to enhance overall performance [30, 31].

Recent advancements in the field have incorporated deep learning to capture more nuanced semantic relationships and have explored cross-lingual transfer learning to extend WSD capabilities to low-resource languages. Moreover, in Gujarati, very little work has been done so far. Research presented in [21] uses a supervised machine learning approach for Word Sense Disambiguation in Gujarati. It likely involves training a model on a labeled dataset to predict the correct sense of ambiguous words. An overview and discussion of the methodology for implementing Word Sense Disambiguation in the Gujarati language are presented. It focuses on the challenges and potential techniques applicable to the language. In [32], the work proposes a knowledge-based approach that uses a genetic algorithm to perform WSD for Gujarati. The system likely leverages lexical resources and evolutionary computation to identify the correct word sense. The paper presented in [33] focuses on applying a genetic algorithm as the core computational technique to resolve word sense ambiguity in the Gujarati language. It explores how this evolutionary method can optimise the selection of the correct meaning.

3. Supervised Machine Learning Algorithms for Word Sense Disambiguation

This section provides an overview of the main methodologies followed in our study; more specifically, two different supervised machine learning algorithms used for word sense disambiguation in Gujarati. The chosen algorithms, such as DT and RF, were selected because they span a large spectrum of classification strategies ranging from probabilistic and instance-based, to hierarchical and ensemble methods. Furthermore, the pipeline and the concrete steps taken when applying each algorithm to the Gujarati sense-annotated corpus are described, with the aim of setting a common framework upon which comparison can be performed.

3.1 WSD for the Gujarati Language using Decision Tree

WSD resolves lexical ambiguity by looking at the context, which has special importance for the Gujarati language since it contains a high number of polysemous words. Therefore, effective machine translation and sentiment analysis tasks depend primarily on effective WSD in this domain. Below we discuss the DT classifier: this model disambiguates through hierarchical feature splitting [7, 17, 34]. Now let us consider its workflow in more detail.

3.1.1 Feature Extraction

Feature extraction is the first step in the entire process, where the raw text is represented as numerical vectors. We consider a window of neighboring tokens to capture the context of the target word effectively. Being a morphologically rich language, this representation in Gujarati will include the POS tags, word embeddings, and morphological markers like suffixes explicitly. The extracted features are represented as a feature vector: $V = [v_1, v_2, \dots, v_n]$, where v_i are features associated with the context. For example, consider the word “દિવસ" (Divasa), which can mean “day” or “festival” depending on the context. For the sentence: “આજનો દિવસ

સુંદર છે." (Ājanō divasa 733undara chē. / Today's day is beautiful.) Features might include: V = [Neighbouring words: {"આજનો", "સુંદર"}, POS tags: {"Determiner", "Adjective"}].

3.1.2 Building and Training the Decision Tree

The DT classifier learns from a training dataset $D = \{(V_i, h_i)\}$, where V_i context features are represented, and h_i is the correct sense of the word is indicated. The tree is built by recursively splitting the data based on features that maximise information gain.

$$\text{Information Gain} = H(S) - \sum_{i=1}^{k} \frac{|S_i|}{|S|} H(S_i) \quad (1)$$

Where, $H(S)$ is the entropy before splitting.

S_i are subsets created after the split.

Entropy is calculated using:

$$H(S) = -\sum_{i=1}^{n} p_i \log_2(p_i) \quad (2)$$

Where p_i is the probability of class i in subset S.

For example, "દિવસ" the DT may split based on the presence of words like "હોળી" (Hōḷī / Holi) or "આજનો" (Ājanō / today). The split that results in the most reduction in entropy is chosen.

3.1.3 Decision Making using Classification

Once the DT is trained, it classifies ambiguous words using decision rules based on the extracted features. Starting from the root node, it traverses down the tree by evaluating conditions at each decision node until reaching a leaf node that predicts the word's sense.

$$\text{Prediction Formula: } \hat{S} = \text{DecisionTree(V)} \quad (3)$$

The tree uses the feature vector V to predict the most likely sense $\hat{S}$.

Gujarati Example: For the sentence "હોળી આનંદના દિવસમાં ઉજવાય છે" (Hōḷī ānandanā divasamāṁ ujavāya chē / Holi is celebrated during the festive day), the DT identifies "હોળી" as a strong contextual clue and predicts the sense of "દિવસ" as "festival."

3.1.4 Evaluation and Refinement

Accuracy, recall, f1-score, and precision are some of the metrics used in evaluating the performance of the model. If the performance is unsatisfactory, the feature set can be adjusted, or the tree depth and splitting criteria can be optimized.

The algorithmic steps for DT execution are shown here for a more detailed understanding. Algorithm 1 describes the execution steps of Gujarati WSD using DT shown below.

Algorithm 1. Gujarati WSD using DT

Input:
Dataset containing columns - Sentence, Target word, Sense
Output:
Corresponding LabelEncoders for each model
Begin
1. Load dataset D.
2. Extract the set of all sentences S from D.
3. Train a Word2Vec (CBOW) model using the pre-processed sentences S with parameters (vector_size=100, window=5, min_count=1, sg=0).
4. Save W2V to disk.
5. Initialize empty main dictionary M = {}, E = {}.
6. Create set T of all unique target words from D.
7. For each target word t_i ∈ T do:
   7.1 Extract subset D_i ⊂ D containing only rows where Target word = t_i.
   7.2 For each sentence in D_i get its vector v_j using W2V
   7.3 Stack v_j into feature matrix X_i; extract label vector y_i (Sense column).
   7.4 Encode the target labels (y_i) using LabelEncoder (LE_i) to convert them into numeric format(y_i_enc). Store E[t_i] = LE_i.
   7.5 Split X_i, y_i_enc into training and validation sets:
      7.5.1Split the data into training (X_train_i)(80%) and validation (X_val_i) (20%) sets.
      7.5.2Train DecisionTree classifier (DT_i) on X_train_i_, y_train_i.
      7.5.3Evaluate DT_i on X_val_i and compute and print validation accuracy.
   7.6 Store the trained model (DT_i), LabelEncoder (LE_i) in the model's dictionary M[t_i] with the target word as the key.
8. After training all target words(t_i), save the models (M).
End

3.2 WSD for the Gujarati Language using Random Forest

An approach to WSD that employs the RF classifier can make reasonably accurate predictions by leveraging ensemble learning. RF is composed of hundreds of DTs, each of which is trained on a random subset of the given characteristics and data. This diversity in training ensures a resilient model that is less prone to overfitting. During the WSD process, context characteristics such as surrounding words, POS tags, and semantic embeddings are first extracted during the feature extraction stage. These characteristics are encoded as numerical vectors and act as inputs to the RF model [7]. The algorithm learns to map different feature patterns to the correct meaning of the word during training. Every DT in the RF casts a vote for a possible interpretation of the word; the majority vote determines the final prediction. The operational methodology of the WSD approach for Gujarati utilizing RF is outlined in detail in the following steps.

3.2.1 Feature Extraction

Here, we extract contextual features of the ambiguous word. These include features related to words, POS tags, and syntactic dependencies. We use morphological features including suffixes and language-specific word embeddings for Gujarati. Each word and its context are represented as a feature vector $V = [v_1, v_2, \ldots, v_n]$, where v_i are features. For

example, consider the word “વર્ગ" (Varga), which can mean “class” or “category” depending on the context. For the sentence: “પાઠશાળાના વિદ્યાર્થીઓ તેમના વર્ગમાં બેસી ગયા." (Pāṭhaśāḷānā vidyārthī’ō tēmanā vargamāṁ bēsī gayā. / School students sat in their respective classes.) Features might include: V = [Neighbouring words: {“વિદ્યાર્થીઓ (Vidyārthī’ō / Students)", “શાળા (Śāḷā / School)"}, POS tags: {“Noun”, “Noun”}, Morphological clues: The word વર્ગમાં (Vargamāṁ / In class) contains suffix “માં" (in)].

3.2.2 Dataset Preparation and Labelling

A labelled dataset $D = \{(\mathrm{V}_i, \mathrm{h}_i)\}$ is created, where V_i represents the feature vector, and h_i is the correct sense label of the ambiguous word.

$$\mathrm{H}_i = \mathrm{argmax}_j \left(P\left(S_j | \mathrm{V}_i\right) \right) \tag{4}$$

Here, $P\left(S_j | \mathrm{V}_i\right)$ is the probability of sense S_j given the feature vector V_i.

For example, sentence 1: “શિક્ષકે નવી પુસ્તકના પાઠ વર્ગમાં શીખવ્યા.” (Śikṣakē navī pustakanā pāṭha vargamāṁ śīkhavyā. / The teacher taught the lesson of the new book to the class.) → h =”class”. Sentence 2: “ખાતાકીય સેવાઓ માટે બેંકે અલગ વર્ગ ઉભા કર્યા.” (Khātākīya sēvā’ō māṭē bēṅkē alaga varga ubhā karyā. / Banks have created a separate category for departmental services.) → h =”category”.

3.2.3 Training the RF Model

The features from the labeled dataset are input to the RF classifier. The algorithm generates multiple DTs, each trained on a randomly selected subset of data and features. The final model combines the predictions of these trees to produce robust, accurate classifications. The RF classifier consists of k DTs. Each tree Ti is trained on a random subset of the training dataset. The prediction formula for a single tree:

$$j_i(\mathrm{V}) = \mathrm{argmax}_j (P(S_j | \mathrm{V})) \tag{5}$$

RF prediction combines the outputs of all trees to make a final prediction:

$$J(\mathrm{V}) = \mathrm{argmax}_j \left(\frac{1}{k} \sum_{i=1}^{k} I(j_i(\mathrm{V}) = S_j) \right) \tag{6}$$

For example, for the word “વર્ગ”, the classifier learns that the presence of words like “વિદ્યાર્થીઓ” (Vidyārthī’ō / students) or “પાઠશાળા” (PāṭhaŚāḷā / school) in the context likely indicates the sense “class” while words like “કંપની” (Kampanī / company) or “સેવાઓ” (Sēvā’ō / services) suggest “category.”

3.2.4 Sense Disambiguation using RF

When a new Gujarati sentence is presented, the trained model analyzes the DT output in the RF to extract its features and predict the senses of ambiguous words. For every sense S_j, the model calculates contextual probabilities and chooses the one with the highest score:

$$\hat{h} = \text{argmax}_j \left(\frac{1}{k} \sum_{i=1}^{k} I(j_i(f(V)) = S_j) \right) \quad (7)$$

Where V represents the input context of the ambiguous word (feature vector), derived from feature extraction. F(V) is the feature extraction function processes the raw input text (including contextual words, POS tags, embeddings, etc.) to create the feature vector.

j_i is the prediction function of the ith DT in the RF is defined here.

k is the total number of DTs in the RF.

I is an indicator function, which outputs 'one' if the predicted sense $j_i(V)$ matches the sense S_j, and 'zero' otherwise.

Argmax$_j$ selects the sense S_j with the highest aggregated score across all trees.

$\hat{h}$ is the final predicted sense for the ambiguous word.

For example, given the sentence "કંપનીએ માર્કેટિંગ માટે અલગ વર્ગ બનાવ્યા." (Kampanī'ē mārkēṭiṅga māṭē alaga varga banāvyā / The company created a separate division for marketing), the model predicts the sense of "વર્ગ" as "category" based on contextual features like "કંપની" and "અલગ.".

3.2.5 Evaluation and Refinement

The model's performance is evaluated using metrics such as accuracy, recall, f1-score and precision. Based on the results, further refinements are made by adjusting the feature set or optimizing hyperparameters.

For a more detailed understanding, the algorithmic steps for executing RF are shown here. Algorithm 2 represents the execution steps of Gujarati WSD using RF as shown below.

Algorithm 2. Gujarati WSD using RF

Input:
Dataset containing columns - Sentence, Target word, Sense
Output:
Trained Random Forest models for each target word
Begin
1. Load dataset D.
2. Extract the set of all sentences S from D.
3. Train a Word2Vec (CBOW) model using the pre-processed sentences S with parameters (vector_size=100, window=5, min_count=1, sg=0).

4. Save W2V to disk.
5. Initialize two empty dictionaries M = {} and E = {}
6. Create set T of all unique target words from D.
7. For each target word t_i ∈ T do:
   7.1 Extract subset D_i ⊂ D containing only rows where Target word = t_i.
   7.2 For each sentence in D_i get its vector v_j using W2V
   7.3 Compute the sentence vector v_j from token vectors v_j = mean({W2V[w] for w in tokens_j ∩ vocab_w}).
   7.4 Stack v_j into feature matrix X_i; extract label vector y_i (Sense column).
   7.5 Encode the target labels (y_i) using LabelEncoder (LE_i) to convert them into numeric format(y_i_enc). Store E[t_i] = LE_i.
   7.6 Split X_i, y_i_enc into training and validation sets:
      7.6.1 Split the data into training (X_train_i)(80%) and validation (X_val_i) (20%) sets. (use stratify when y_i_enc has ≥2 classes).
      7.6.2 Train Random Forest classifier (DT_i) on X_train_i_, y_train_i.
      7.6.3 Evaluate RF_i on X_val_i and compute and print validation accuracy.
   7.7 Store the trained model (RF_i), LabelEncoder (LE_i) in the model's dictionary M[t_i] with the target word as the key.
8. After training all target words(t_i), save the models (M).

End

4. A Model Training and Testing Framework for Supervised Gujarati WSD

Building reliable and accurate WSD systems for Gujarati requires a systematic approach to model development and evaluation. This section outlines the end-to-end framework adopted for training supervised learning models on the sense-annotated Gujarati corpus and rigorously evaluating their performance. The framework incorporates two widely adopted machine learning algorithms DT and RF, representing a breadth of hierarchical and ensemble techniques. The proposed framework ensures that each classifier is trained on representative data and subsequently assessed on previously unseen examples to validate its ability to resolve lexical ambiguity across diverse contexts. The following subsections describe in detail the procedures and considerations involved in the training phase, where models learn to associate contextual features with word senses, and the testing phase, where generalization capabilities and predictive accuracy are quantified.

4.1 Training Procedure for Supervised Gujarati WSD

The flowchart in Figure 4 illustrates the structured pipeline for training a collection of supervised machine learning models for WSD in Gujarati. The procedure begins with data ingestion and preprocessing, proceeds to feature representation using word embeddings, and concludes with the training of two separate classifiers. Every step is crucial for converting unrefined textual material into a structured format that allows models to comprehend the contextual subtleties necessary for precise sense disambiguation.

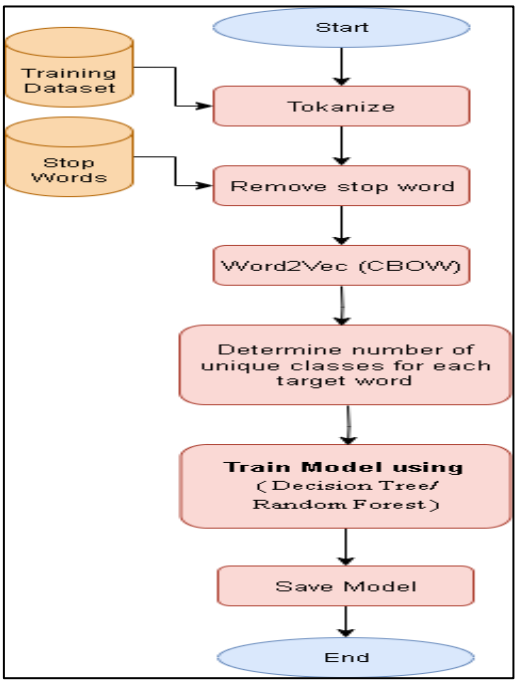


Figure 4. Training Module for Gujarati WSD using Supervised Algorithms

4.1.1 Data Input and Preprocessing

- Training Dataset and Stop Words: The workflow commences with two primary inputs: the Training Dataset and a list of Stop Words. The training dataset is a corpus of Gujarati sentences, with ambiguous words manually annotated with their correct sense labels. The stop words list contains common Gujarati words (e.g., "છે" (chhe/is), "અને" (ane/and), "એક" (ek/a)) that carry little semantic weight and are subsequently removed to reduce noise.

- Tokenization: The first step in the WSD process is Tokenization. In this stage, each sentence from the training dataset is segmented into a sequence of individual words or "tokens." This fundamental step breaks unstructured text into discrete units that can be processed individually. Each Gujarati sentence was tokenized using a whitespace-based tokenizer after converting the text to lowercase and removing non-alphabetic characters. It also eliminates stopwords when applicable. Only valid tokens present in the Word2Vec vocabulary were retained. The final sentence vector was computed as the average of all word embeddings corresponding to the retained tokens. For example, the sentence "મેચમાં ભારતની જીત થઈ" (Mēcamāṁ Bhāratnī jīta thaī / India won the match) would be tokenized into ['મેચમાં', 'ભારતની', 'જીત', 'થઈ'].

- Stop Word Removal: After tokenization, the predefined Gujarati stop words list is used to filter the tokenized text. Each token is compared against the stop word list, and if a match is found, the token is discarded. This ensures that the subsequent feature extraction process focuses only on contextually significant words. For

example, if "થઈ" (thaī) is in the stop word list, the token list from the previous step becomes ['મેચમાં', 'ભારતની', 'જીત'].

4.1.2 Feature Representation and Class Identification

- **Word2Vec:** The purpose of the Feature Extraction step is to generate and store word embeddings using the Word2Vec model, providing structured numerical representations that can be used as input features for the WSD classifier. To represent the semantic meaning of the processed tokens numerically, the Word2Vec model is employed, explicitly using the Continuous Bag-of-Words (CBOW) architecture. This model transforms each word into a high-dimensional vector (embedding) that captures its contextual relationships with other words in the corpus. The CBOW architecture is particularly effective because it predicts a target word based on its surrounding context words, thereby generating embeddings rich in semantic information. These vectors serve as the primary features for the machine learning classifiers.

- **Determine Number of Unique Classes:** Concurrently, the system identifies the number of unique senses (classes) for each target ambiguous word in the dataset. This step is crucial for defining the scope of the classification task for each word. The model needs to know precisely how many possible meanings it must choose from. For example, for the target word "હાર" (Hār), the system would identify two unique classes from the annotated dataset: sense-1 (પરાજય/Defeat) and sense-2 (ઘરેણું/Necklace). This defines the problem as a two-class classification task.

4.1.3 Model Training

- **Train Model:** This is the core stage of the workflow, where the prepared data (contextual word embeddings as features and sense labels as target classes) is used to train two distinct supervised machine learning algorithms, such as: 1. Decision Tree: A hierarchical model that makes decisions based on a series of rules learned from the data. 2. Random Forest: An ensemble method that constructs multiple DTs and outputs the mode of their predictions, often leading to higher accuracy and robustness. Each of these models is trained independently on the same dataset to learn the patterns that correlate specific contextual vectors with their corresponding word senses.

- **Save Model:** Once a model has been successfully trained, it is saved to disk. This final step ensures that the trained classifier can be easily loaded and used for testing or deployed in a practical application without requiring the entire training process to be repeated.

4.2 Testing Procedure for Supervised Gujarati WSD

The testing module, illustrated in Figure 5, is designed to evaluate the performance and adaptation capabilities of previously trained supervised models on novel data. This stage is crucial for validating the effectiveness of the Word Sense Disambiguation (WSD) system. The workflow methodically examines new Gujarati phrases, extracts pertinent features, and

employs a trained classifier to determine the correct meaning of an ambiguous word, replicating the preprocessing pipeline from the training phase to maintain feature consistency.

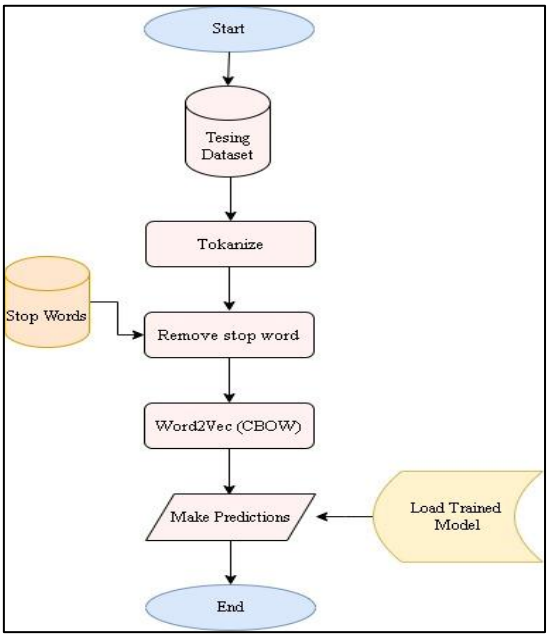


Figure 5. Testing Module for Gujarati WSD using Supervised Algorithms

4.2.1 Data Input and Preprocessing

- **Testing Dataset:** The testing dataset comprises Gujarati sentences that were not used during model training. Each sentence in this dataset contains an instance of an ambiguous word for the model to predict the actual sense.
- **Tokenization and Stop Word Removal:** The input sentence undergoes the same initial preprocessing steps as in the training module. First, it undergoes tokenization, in which the sentence is segmented into a list of individual words (tokens). Subsequently, stop word removal is performed by filtering out common, low-information Gujarati words (e.g., "અને" (and), "માટે" (for)) using a predefined stop word list. This ensures that the feature set is focused on the most semantically significant contextual words. For example: an input test sentence, “રાણીએ ગળામાં હીરાનો હાર પહેર્યો હતો.” (Rāṇī'ē gaḷāmāṁ hīrānō hār pahēryō hatō / The queen wore a diamond necklace), tokenization would yield `['રાણીએ', 'ગળામાં', 'હીરાનો', 'હાર', 'પહેર્યો', 'હતો']`. After stop word removal (assuming 'હતો' is a stop word), the list would be reduced to `['રાણીએ', 'ગળામાં', 'હીરાનો', 'હાર', 'પહેર્યો']`.

4.2.2 Feature Extraction and Prediction

- **Word2Vec:** The preprocessed tokens are then transformed into numerical feature vectors using the same pre-trained CBOW model used during training. This step is crucial for maintaining consistency in the vector space representation between the

training and testing data. Each context word is converted into a high-dimensional embedding that captures its semantic meaning.

- **Load Trained Model:** In parallel, a trained model (e.g., the saved RF) is loaded from memory. This loaded model contains the learned patterns and decision boundaries derived from the training dataset.
- **Make Predictions:** This is the concluding step of the testing process. The feature vectors generated from the test sentence are fed into the loaded classifier. The model then applies its learned logic to these input features to make a prediction, assigning the most probable sense to the ambiguous word based on its context. For example, the loaded model would analyze the vectors for "રાણીએ", "ગળામાં", and "હીરાનો" as the context for the ambiguous word "હાર". Based on the patterns it learned during training, it would predict the sense as "ઘરેણું" (Gharēṇuṁ/Jewellery), not "પરાજય" (Parājaya/Defeat). The predicted sense is the final output of the testing workflow for a given sentence. Then we compute various performance metrics, such as accuracy, precision, recall, and F1-score, thereby quantifying the model's effectiveness.

5. Results and Discussion

The empirical results of our comparative assessment of Decision Tree and Random Forest classifiers for Gujarati WSD are presented in this chapter. We begin by defining the experimental framework that details the construction of our custom-annotated dataset and the metrics used to validate the results.

5.1 Experimental Setup and Dataset

The performance of the DT and RF algorithms was evaluated on a novel, manually annotated Gujarati corpus developed specifically for this study. The corpus was designed to be a controlled environment for WSD. It consists of 1,084 unique sentences built around a curated set of 50 ambiguous Gujarati words with 113 different senses. All the experiments were conducted using a standard desktop computer. Its computational resource requirements regarding this methodology are modest; we found a standard quad-core CPU and 8 GB of RAM are sufficient for all data processing, model training, and testing phases. The experiments were conducted using 3-fold and 5-fold cross-validation to evaluate Gujarati WSD performance under two conditions: (1) with stop words and (2) after removing stop words. For each fold configuration, the dataset was divided into k equal-sized subsets. At each iteration, one subset served as the test set, and the remaining (k–1) subsets formed the training set. Thus, the 3-fold setup took approximately 66.6% for training and 33.3% for testing at each fold, while the 5-fold setup used 80% for training and 20% for testing at each fold. During training, the models analyzed the features of a sentence and learned the patterns of each word sense from the labels created by human annotators. The rest (based on fold values) was kept separate and used in testing the models after their training. By evaluating the models on data they have never seen before, we can obtain an unbiased measure of how accurately the algorithms can disambiguate words in new, unseen Gujarati sentences. This demonstrates how well the models generalize

their knowledge. An example of a reference dataset is outlined in Table 2. The ensuing section discusses the results obtained from the classifiers used in this dataset.

Table 2. Example of Gujarati Dataset for WSD

Sentence	Target Word	Sense
હું તને બોલાવવા ખાતર ખાસ આવ્યો હતો. (Huṁ tanē bōlāvavā khātara khāsa āvyō hatō. / I came specifically to call you.)	ખાતર (Khātara)	માટે (Māṭē / for)
તેના મકાનમાં થયેલી ખાતરથી લોકો ચિંતિત હતા. (Tēnā makānamāṁ thayēlī khātarathī lōkō cintita hatā. / People were worried about the damage done to his house.)		ચોરી (Cōrī / theft)
ખેતરમાં યોગ્ય માત્રામાં ખાતરનો ઉપયોગ કરવામાં આવ્યો. (Khētaramāṁ yōgya mātrāmāṁ khātaranō upayōga karavāmāṁ āvyō. / The right amount of fertilizer was used in the field.)		ખેતરનું ખાતર (Khētaranuṁ khātara / fertilizer)
બાળકોને ઘોડો પાર સવારી કરવાની માજા આવે છે (Bāḷakōnē ghōḍō pāra savārī karavānī mājā āvē chē / Children enjoy riding horses.)	ઘોડો (Ghōḍō)	પ્રાણી (Prāṇī / Animal)
મમ્મી એ રસોડાના ઘોડા પર વાસણ ગોઠવ્યા. (Mam'mī ē rasōḍānā ghōḍā para vāsaṇa gōṭhavyā. / Mom arranged the dishes on the kitchen counter.)		ગોઠવણનું સાધન (Gōṭhavaṇanuṁ sadhana / Adjustment tool)
અમારા ઘરના માળિયામાં ઘોડો ની મદદ થી જ બધી વસ્તુઓ મૂકી શકાય છે. (Amārā gharanā māḷiyāmāṁ ghōḍō nī madada thī ja badhī vastu'ō mūkī śakāya chē. / All the things in our house's attic can be moved only with the help of a wooden easel.)		લાકડાની ઘોડી (Lākaḍānī ghōḍī / Wooden easel)
ભગવાનના ગળામાં આજે મોગરાનો હાર ધરાવેલો હતો. (Bhagavānanā gaḷāmāṁ ājē mōgarānō hāra dharāvēlō hatō.	હાર (Hāra)	ફૂલમાળા (Phūlamāḷā / Flower Garland)

/ Today, the Lord wore a garland of flowers “Mogra” around his neck.)		
ક્રિકેટ ની ટીમ માં એક ટીમ એ હાર સ્વીકારી લીધી. (Krikēṭa nī ṭīma māṁ ēka ṭīma ē hāra svīkārī līdhī. / One of the cricket teams accepted defeat.)		પરાજય (Parājaya / Defeat)
સ્પર્ધામાં ભાગ લેવા માટે એક હારમાં ઉભા રહો. (Spardhāmāṁ bhāga lēvā māṭē ēka hāramāṁ ubhā rahō. / Stand in a row to participate in the competition.)		પંક્તિ (Paṅkti/ Row)
તારો સોનાનો હાર ખુબ જ સુંદર છે. (Tārō sōnānō hāra khuba ja sundara chē. / Your gold necklace is very beautiful.)		ઘરેણું (Gharēṇuṁ / Jewelry)

5.2 Performance Comparison of DT and RF Supervised Algorithms for Gujarati WSD

This section describes the performance evaluation of Gujarati WSD using DT and RF models, carried out under two controlled conditions: with stop words and without stop words, and across 3-fold and 5-fold cross-validation settings. The detailed numerical results are presented in Tables 3, 4, 5, and 6, whereas their graphical depictions appear in Figures 6 to 9. In addition, the performance of the Gujarati WSD was evaluated in terms of accuracy, precision, recall, and F1-score. Results are also reported using Mean ± Standard Deviation (SD).

Table 3. Parametric Evaluation (Gujarati WSD using DT With and Without Stop Words) for 5 Folds

Gujarati WSD using DT with Stop Words for 5-Fold					**Gujarati WSD using DT without Stop Words for 5-Fold**			
	Measured Value (In %)				Measured Value (In %)			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Fold1	50.69	45.55	49.63	45.24	48.85	44.13	47.49	43.03
Fold2	49.77	42.5	49.03	43.36	45.16	40.59	44.62	39.59
Fold3	48.85	44.97	47.62	43.24	47	41.41	43.45	40.28
Fold4	50.46	46.79	51.41	46.64	49.07	43.63	49.03	43.29

Fold5	50.93	45.53	49.59	45.3	48.15	41.21	47.04	41.33
Mean	50.14	45.07	49.46	44.76	47.65	42.19	46.33	41.50
Standard Deviation	0.84	1.58	1.36	1.44	1.61	1.58	2.26	1.64

Table 4. Parametric Evaluation (Gujarati WSD using DT With and Without Stop Words) for 3 Folds

Gujarati WSD using DT with Stop Words for 3-Fold					**Gujarati WSD using DT without Stop Words for 3-Fold**			
	Measured Value (In %)				Measured Value (In %)			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Fold1	45.71	43.81	44.07	41.41	51.25	48.58	49.74	46.42
Fold2	45.15	43.64	42.47	40.3	45.15	44.23	45.96	42.53
Fold3	48.48	45.82	47.63	45.14	47.92	46.53	46.96	44.34
Mean	46.45	44.42	44.72	42.28	48.11	46.45	47.55	44.43
Standard Deviation	1.78	1.21	2.64	2.54	3.05	2.18	1.96	1.95

Table 5. Parametric Evaluation (Gujarati WSD using RF With and Without Stop Words) for 5 Folds

Gujarati WSD using RF with Stop Words for 5-Fold					**Gujarati WSD using RF without Stop Words for 5-Fold**			
	Measured Value (In %)				Measured Value (In %)			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Fold1	54.38	49.60	52.21	48.60	50.69	42.79	47.79	42.71
Fold2	56.68	52.81	55.56	51.04	55.30	51.40	55.26	50.48
Fold3	56.68	50.96	55.18	50.38	52.07	48.14	50.15	46.21

Fold4	58.80	52.46	58.93	52.36	54.63	49.79	54.61	48.85
Fold5	58.33	51.89	57.59	52.66	58.33	49.91	57.50	50.48
Mean	56.97	51.54	55.89	51.01	54.20	48.41	53.06	47.75
Standard Deviation	1.74	1.29	2.56	1.64	2.97	3.34	3.97	3.31

Table6. Parametric Evaluation (Gujarati WSD using RF With and Without Stop Words) for 5 Folds

Gujarati WSD using RF with Stop Words for 3-Fold					**Gujarati WSD using RF without Stop Words for 3-Fold**			
	Measured Value (In %)				Measured Value (In %)			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Fold1	55.96	53.93	53.32	51.08	56.79	55.30	55.39	52.25
Fold2	55.12	56.89	53.37	51.23	51.25	47.32	48.31	45.70
Fold3	59.56	58.15	58.06	54.27	53.46	51.95	51.98	48.52
Mean	56.88	56.32	54.92	52.19	53.83	51.52	51.89	48.82
Standard Deviation	2.36	2.17	2.72	1.80	2.79	4.01	3.54	3.29

The random forest results for 5-fold with stop words were significantly better than DT: 56.97 ± 1.74 accuracy, 51.54 ± 1.29 precision, 55.89 ± 2.56 recall, and 51.01 ± 1.64 f1-Score. RF performance decreased without stop words to 54.20 ± 2.97 accuracy, 48.41 ± 3.34 precision, 53.06 ± 3.97 recall, and an F1-score of 47.75 ± 3.31. Similarly, in the case of the 3-fold evaluation, RF with stop words yielded 56.88 ± 2.36 accuracy, 56.32 ± 2.17 precision, 54.92 ± 2.72 recall, and an F1-score of 52.19 ± 1.80, whereas its stop-word-removed variant achieved just 53.83 ± 2.79 accuracy, 51.52 ± 4.01 precision, 51.89 ± 3.54 recall, and 48.82 ± 3.29 F1-score. RF clearly demonstrates more stable performance, as can be seen by the lower standard deviation in all metrics. These are summarized in Figure 6.

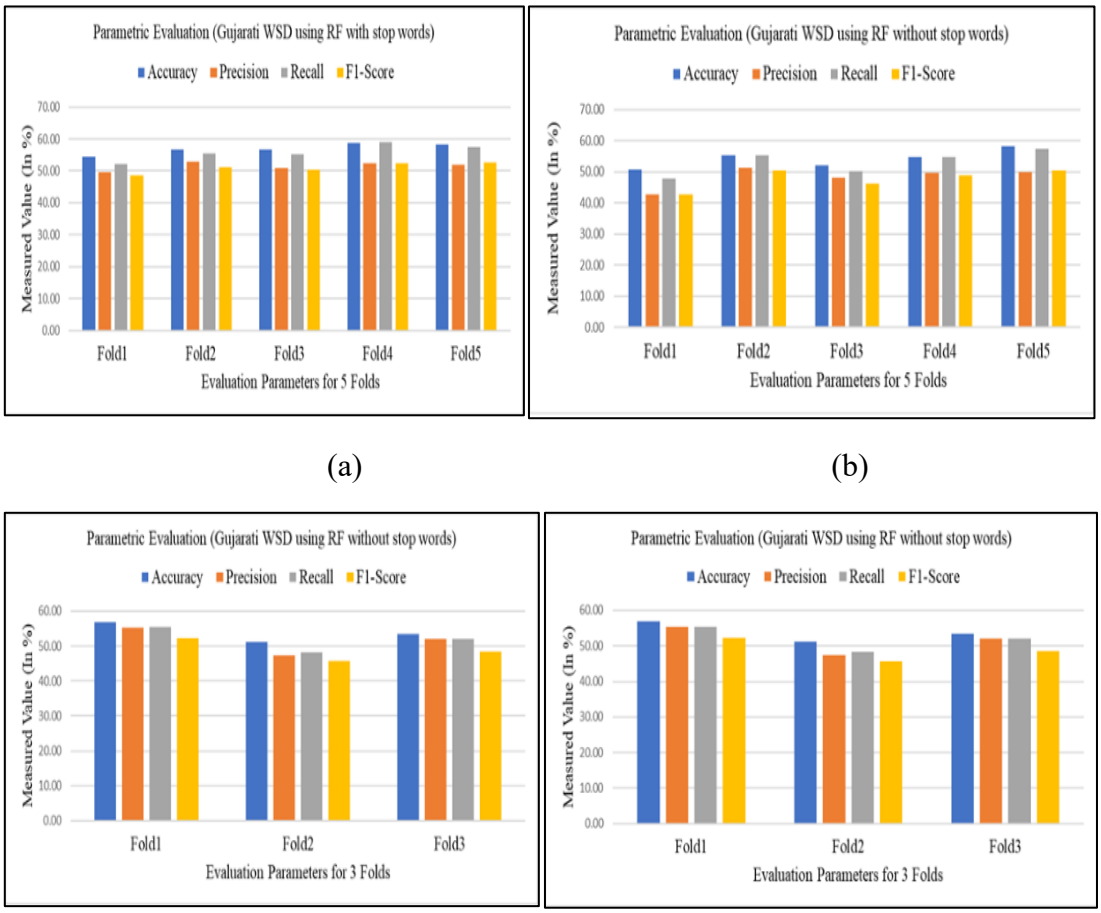


Figure 6. (a) Parametric Evaluation (Gujarati WSD using RF With Stop Words) for 5 Folds (b) Parametric Evaluation (Gujarati WSD using RF Without Stop Words) for 5 Folds (c) Parametric Evaluation (Gujarati WSD using RF With Stop Words) for 3 Folds (d) Parametric Evaluation (Gujarati WSD using RF Without Stop Words) for 3 Folds

For the 5-fold Decision Tree, with stop words, the model achieved an accuracy of 50.14 ± 0.84, precision of 45.07 ± 1.58, recall of 49.46 ± 1.36, and an F1-score of 44.76 ± 1.44. Without stop words, performance decreased to 47.65 ± 1.61 accuracy, 42.19 ± 1.58 precision, 46.33 ± 2.26 recall, and 41.50 ± 1.64 F1-score, which indicated that DT benefits from keeping stop words that carry syntactic or contextual cues useful for disambiguation. In the case of a 3-fold evaluation, DT with stop words obtained 46.45 ± 1.78 accuracy, 44.42 ± 1.21 precision, 44.72 ± 2.64 recall, and an F1-score of 42.28 ± 2.54. After removing the stop words, the performance slightly improved to 48.11 ± 3.05 accuracy, 46.45 ± 2.18 precision, 47.55 ± 1.96 recall, and an F1-score of 44.43 ± 1.95. Hence, it seems that, for smaller training sets such as those in the 3-fold configuration, reducing the lexical noise has had a positive impact. These results are summarized in Figure 7.

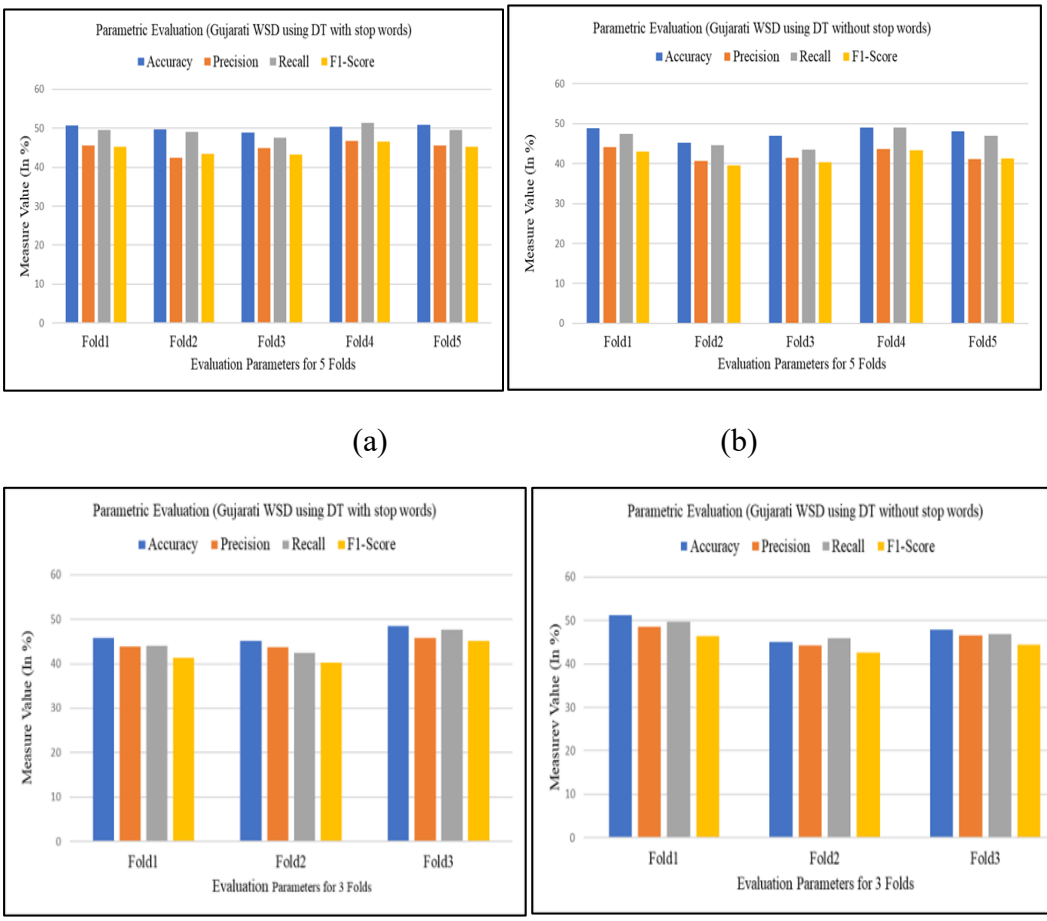
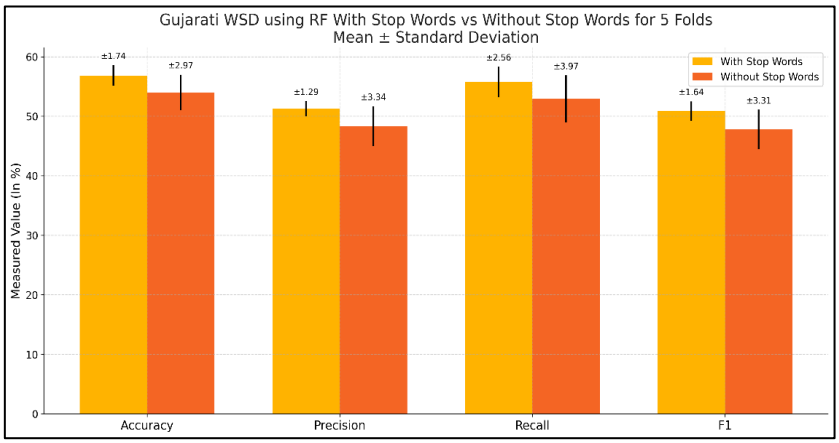
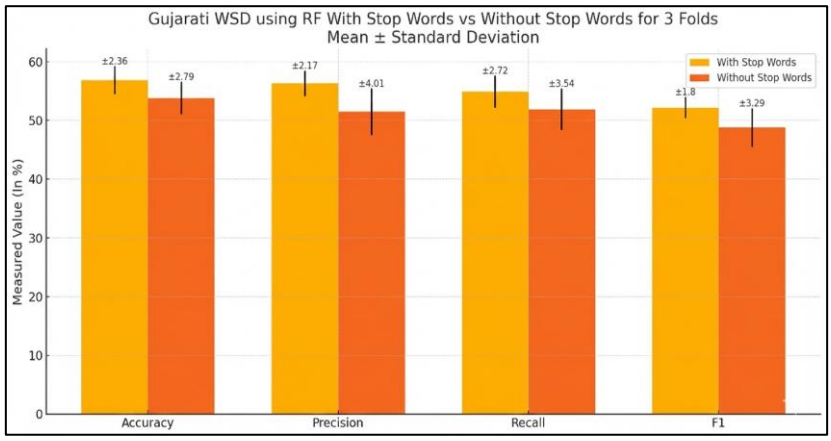


Figure 7. (a) Parametric Evaluation (Gujarati WSD using DT With Stop Words) for 5 Folds (b) Parametric Evaluation (Gujarati WSD using DT Without Stop Words) for 5 Folds (c) Parametric Evaluation (Gujarati WSD using DT With Stop Words) for 3 Folds (d) Parametric Evaluation (Gujarati WSD using DT Without Stop Words) for 3 Folds

Statistical analysis using error bars Mean ± SD demonstrates that RF with stop words makes the most consistent and stable predictions among all, as reflected in Figure 8, whereas for DT and RF without stop words, the slightly higher SD values across different folds indicate larger variability. In the case of RF, lower SD values further support its robustness. DT shows mixed responses to stop-word removal across different folds, as depicted in Figure 9, while RF consistently improves with stop words. Comparative analysis reveals that Random Forest consistently outperforms Decision Tree across all metrics and different sets of evaluation. RF demonstrates more robustness, higher accuracy, better generalization, and lower variability. DT is more sensitive to the changes in training size and therefore suffers from higher variance due to its simpler model structure. Regarding computational performance, both models operate on precomputed sentence embeddings. Since this is a feature extraction one-time cost, it does not add to the latency in prediction. Therefore, for the sentence-level WSD task, both the Decision Tree and Random Forest models are computationally efficient, with the difference in their prediction latency being negligible. Both models were susceptible to the nature of Gujarati stop words, which have syntactic and semantic clues, but RF made better use of this additional information.



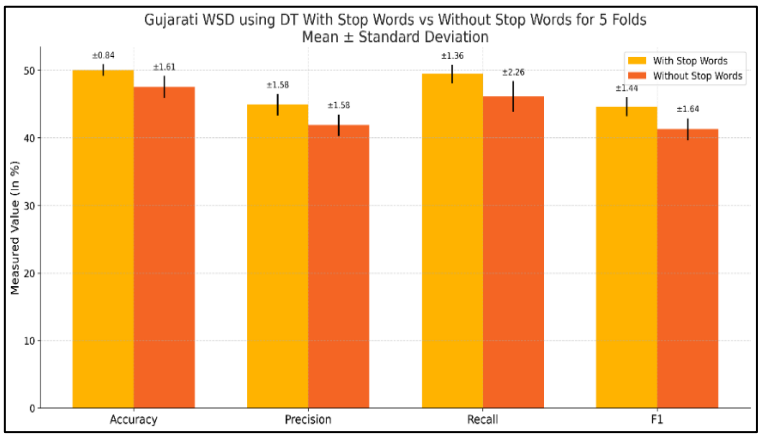
(a)



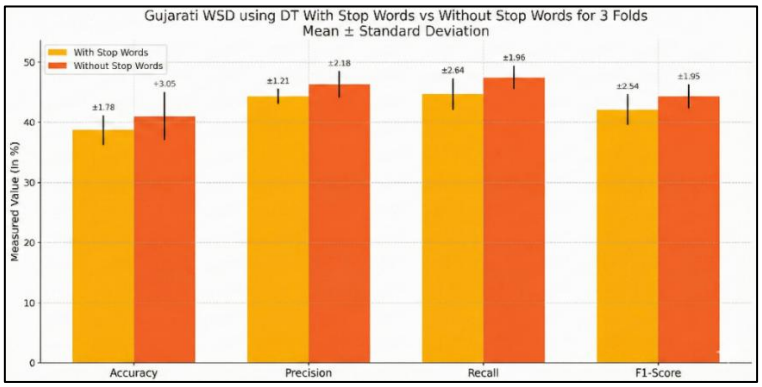
(b)

Figure 8. (a) Mean ± Standard Deviation of Gujarati WSD using RF over 5 Folds, With and Without Stop Words (b) Mean ± Standard Deviation of Gujarati WSD using RF over 3 folds, With and Without Stop Words

The RF algorithm, therefore, is the best result produced for Gujarati WSD. The results strongly indicate that keeping stop words as features remains useful since they provide essential contextual clues, greatly improving the accuracy of the more complex ensemble model.



(a)



(b)

Figure 9. (a) Mean ± Standard Deviation of Gujarati WSD using DT over 5 Folds, With and Without Stop Words (b) Mean ± Standard Deviation of Gujarati WSD using DT over 3 Folds, With and Without Stop Words

6. Conclusion and Future Directions

This research paper effectively addresses the issue of Gujarati word sense disambiguation and achieves its primary aims while answering all essential research questions. Besides fulfilling the primary research aims, this work introduces the Guj-Sense corpus, demonstrating the feasibility of creating high-quality resources for low-resource languages. The validation of this corpus as a robust benchmark for supervised learning confirms the possibility of creating such a resource. Further testing provides clear evidence of which models yield the best performance. The performance comparison of algorithms on this dataset shows that Random Forest significantly outperforms Decision Trees in all dimensions of performance evaluated. This significant edge in performance indicates that ensemble approaches are far better equipped to model the lexical ambiguity inherent in morphologically complex languages such as Gujarati. Furthermore, the data demonstrate that stop words are essential contextual cues for robust models, as retaining them contributes to measurable increases in accuracy.

Future research will investigate scaling the Guj-Sense corpus towards a larger vocabulary, more diverse domains, and regional dialects. Going beyond traditional classifiers, the deep learning paradigm can be used to test BiLSTMs and Transformer-based models. We will also research whether including fine-grained linguistic features, such as syntactic dependency relations and richer POS-tagging, yields further improvements in accuracy. A useful extension of this work could involve a systematic study of how different contextual window sizes—e.g., words ±2, ±5, ±10—affect the performance of Gujarati WSD. Although the present study demonstrates that DT and RF models perform well for Gujarati WSD, it lacks formal analysis regarding feature importance.

References

[1] Dave, Avani N., and Sanjay M. Shah. "Comprehensive Analysis for Assessing the Effectiveness in the Implementation of Word Sense Disambiguation in the Gujarati Language." In IET Conference Proceedings CP920, vol. 2025, no. 7, Stevenage, UK: The Institution of Engineering and Technology, 2025, 1093-1100.

[2] Navigli, Roberto. "Word Sense Disambiguation: A Survey." ACM computing surveys (CSUR) 41, no. 2 (2009): 1-69.

[3] Gujjar, Vinto, Neeru Mago, Raj Kumari, Shrikant Patel, Nalini Chintalapudi, and Gopi Battineni. "A Literature Survey on Word Sense Disambiguation for the Hindi Language." Information 14, no. 9 (2023): 495.

[4] Escudero Bakx, Gerard. "Machine Learning Techniques for Word Sense Disambiguation." PhD diss., Universitat Politècnica de Catalunya (UPC), 2006.

[5] Dave, Nakul R., and Mayuri A. Mehta. "Comparative Analysis of Rule-Based, Dictionary-Based and Hybrid Stemmers for Gujarati Language." In International Conference on Big Data Analytics, Cham: Springer International Publishing, 2019, 140-155.

[6] Sinha, M., Kumar, M., Pande, P., Kashyap, L., & Bhattacharyya, P. (2004). Hindi Word Sense Disambiguation. International Symposium on Machine Translation, Natural Language Processing and Translation Support Systems, 1–7. https://api.semanticscholar.org/CorpusID:9438332

[7] Abraham, Ajith, Bineet Kumar Gupta, Satya Bhushan Verma, Archana Sachindeo Maurya, Mohammad Husain, Arshad Ali, Sami Alshmrany, and Sanjay Gupta. "Improvement of Translation Accuracy for the Word Sense Disambiguation System using Novel Classifier Approach." International Arab Journal of Information Technology (IAJIT) 21, no. 6 (2024).

[8] Zhou, Xiaohua, and Hyoil Han. "Survey of Word Sense Disambiguation Approaches." In FLAIRS, 2005, 307-313.

[9] Hao, Lianwang, Tao Zhang, and Huaixin Liang. "A Novel Rotational Causal Random Forest Approach for Word Sense Disambiguation of English Modal Verbs." Available at SSRN 5244147.

[10] Vasoya, Parth J., and T. Vyas. "A Survey on Word Sense Disambiguation Approaches." International journal of Trend in Research and Development 1 (2014): 1-3.

[11] Rekha, Smt. V. and Manish Shah. “A Study of Main Elements of Word Sense Disambiguation (WSD) for Gujarati Language.” (2016).

[12] Sheth, Mitul, Shivang Popat, and Tarjni Vyas. "Word Sense Disambiguation for Indian Languages." In International conference on emerging research in computing, information, communication and applications, Singapore: Springer Singapore, 2016, 583-593.

[13] Vyas, Tarjni, and Amit Ganatra. "Gujarati Language: Research Issues, Resources and Proposed Method on Word Sense Disambiguation." Int. J. Recent Technol. Eng.(IJRTE) ISSN 8, no. 2 suppl 11 (2019): 2277-3878.

[14] Geleta, Tabor Wegi, and Jara Muda Haro. "Semisupervised Learning-Based Word-Sense Disambiguation Using Word Embedding for Afaan Oromoo Language." Applied Computational Intelligence and Soft Computing 2024, no. 1 (2024): 4429069.

[15] Borah, Pranjal Protim, Gitimoni Talukdar, and Arup Baruah. "Assamese Word Sense Disambiguation Using Supervised Learning." In 2014 International Conference on Contemporary Computing and Informatics (IC3I), IEEE, 2014, 946-950.

[16] Wang, Tinghua, Junyang Rao, and Qi Hu. "Supervised Word Sense Disambiguation Using Semantic Diffusion Kernel." Engineering Applications of Artificial Intelligence 27 (2014): 167-174.

[17] Sarmah, Jumi, and Shikhar Kr Sarma. "Decision Tree Based Supervised Word Sense Disambiguation for Assamese." Int. J. Comput. Appl 141, no. 1 (2016): 42-48.

[18] Kokane, Chandrakant D., and Sachin D. Babar. "Supervised Word Sense Disambiguation with Recurrent Neural Network Model." Int. J. Eng. Adv. Technol.(IJEAT) 9, no. 2 (2019).

[19] Lai, Huei-Ling, Hsiao-Ling Hsu, Jyi-Shane Liu, Chia-Hung Lin, and Yanhong Chen. "Supervised Word Sense Disambiguation on Polysemy with Neural Network Models: A Case Study of BUN in Taiwan Hakka." International Journal of Asian Language Processing 30, no. 03 (2020): 2050011.

[20] Saif, Abdulgabbar, Nazlia Omar, Ummi Zakiah Zainodin, and Mohd Juziaddin Ab Aziz. "Building Sense Tagged Corpus Using Wikipedia for Supervised Word Sense Disambiguation." Procedia Computer Science 123 (2018): 403-412.

[21] Vyas, Tarjni, and Amit Ganatra. "Gujarati Language Model: Word Sense Disambiguation Using Supervised Technique." Int. J. Rec. Technol. Eng 8, no. 2 (2019): 3740-3744.

[22] Kumar, Sailendra, and Rakesh Kumar. "Word Sense Disambiguation in the Hindi Language: Neural Network Approach." Int. J. Tech. Res. Sci 1 (2021): 72-76.

[23] Preeti, B. "Word Sense Disambiguation in Gujarati Language." Int J Innov Res Comput Sci Technol (IJIRCST) 3, no. 1 (2015): 44-47.

[24] Le, Anh-Cuong, Akira Shimazu, Van-Nam Huynh, and Le-Minh Nguyen. "Semi-Supervised Learning Integrated with Classifier Combination for Word Sense Disambiguation." Computer Speech & Language 22, no. 4 (2008): 330-345.

[25] Taghipour, Kaveh, and Hwee Tou Ng. "Semi-Supervised Word Sense Disambiguation Using Word Embeddings in General and Specific Domains." In Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: human language technologies, 2015, 314-323.

[26] Rahman, Nazreena, and Bhogeswar Borah. "An Unsupervised Method for Word Sense Disambiguation." Journal of King Saud University-Computer and Information Sciences 34, no. 9 (2022): 6643-6651.

[27] Klapaftis, Ioannis P., and Suresh Manandhar. "Unsupervised Word Sense Disambiguation Using The WWW." Frontiers in Artificial Intelligence and Applications 142 (2006): 174.

[28] Martinez-Gil, Jorge. "Context-Aware Semantic Similarity Measurement for Unsupervised Word Sense Disambiguation." arXiv preprint arXiv:2305.03520 (2023).

[29] Kwon, Sunjae, Rishabh Garodia, Minhwa Lee, Zhichao Yang, and Hong Yu. "Vision Meets Definitions: Unsupervised Visual Word Sense Disambiguation Incorporatin Gloss Information." arXiv preprint arXiv:2305.01788 (2023).

[30] Pal, Alok Ranjan, Anirban Kundu, Abhay Singh, Raj Shekhar, and Kunal Sinha. "A Hybrid Approach to Word Sense Disambiguation Combining Supervised and Unsupervised Learning." arXiv preprint arXiv:1611.01083 (2015).

[31] Specia, Lucia. "A Hybrid Model for Word Sense Disambiguation in English-Portuguese Machine Translation." In Proceedings of the 8th Research Colloquium of the UK Special interest Group in Computational Linguistics, pp. 71-78. 2005.

[32] Vaishnav, Zankhana B., and Priti S. Sajja. "Knowledge-Based Approach for Word Sense Disambiguation Using Genetic Algorithm for Gujarati." In Information and Communication Technology for Intelligent Systems: Proceedings of ICTIS 2018, Volume 1, Singapore: Springer Singapore, 2018, 485-494.

[33] Vaishnav, Zankhana B. "Gujarati Word Sense Disambiguation Using Genetic Algorithm." International Journal on Recent and Innovation Trends in Computing and Communication 5, no. 6 (2017): 635-639.

[34] Rawat, Sunita, K. Kalambe, G. Kawade, and N. Korde. "Supervised Word Sense Disambiguation Using Decision Tree." International Journal of Recent Technology and Engineering (IJRTE) 8, no. 2 (2019): 4043-4047.

Appendix

List of Notations

Notation	Identifier (Description)
D	The complete input dataset.
S	The set of all sentences from the dataset.
W2V	The trained Word2Vec model.
M	The main dictionary storing all trained models and encoders.
E	The dictionary of LabelEncoders for all target words (encoders).
T	The set of all unique target words.
t_i	A single unique target word (the i-th word from set T).
D_i	The subset of data corresponding to the target word t_i.
v_j	The vector representation for a single sentence (j).
X_i	The feature matrix (all sentence vectors) for target word t_i.
y_i	The original target vector (text labels) for target word t_i.
LE_i	The LabelEncoder for the senses of target word t_i.
y_i_enc	The numerically encoded target vector for target word t_i.
X_train_i	The training subset (e.g., 80%) of the feature matrix for word t_i.
X_val_i	The validation subset (e.g., 20%) of the feature matrix for word t_i.
y_train_i	The training subset of the encoded target vector for word t_i.
y_val_i	The validation subset of the encoded target vector for word t_i.
DT_i	The trained Decision Tree Classifier model for target word t_i.
RF_i	The trained RandomForestClassifier for t_i.
vocab_w	Shorthand for membership test of a word in W2V vocabulary.