**ABSTRACT**

**Wordnet** is a [lexical database](https://en.wikipedia.org/wiki/Lexical_database) for the [English language](https://en.wikipedia.org/wiki/English_language). It groups English [words](https://en.wikipedia.org/wiki/Word) into sets of [synonyms](https://en.wikipedia.org/wiki/Synonyms) called [*synsets*](https://en.wikipedia.org/wiki/Synsets), provides short definitions and usage examples, and records a number of relations among these synonym sets or their members. WordNet can thus be seen as a combination of [dictionary](https://en.wikipedia.org/wiki/Dictionary) and [thesaurus](https://en.wikipedia.org/wiki/Thesaurus). While it is accessible to human users via a [web browser](https://en.wikipedia.org/wiki/Web_browser), its primary use is in automatic [text analysis](https://en.wikipedia.org/wiki/Natural_language_processing) and [artificial intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence) applications.

Gujarati language is the youngest member of IndoWordnet. As a part of IndoWordnet project, Wordnet for Gujarati language is being developed from Hindi Wordnet using expansion approach. This project demonstrates various synsets ,examples and glossary of different gujarati words. We have tried to show various relation between words such as hypernym,hyponym,meronym,antonym,synonym and many more.

Various path similarity functions(Path similarity ,Leacock-Chodorow Similarity ,Wu-Palmer Similarity ,Resnik Similarity ,Jiang-Conrath Similarity, Lin Similarity) are partly implemented in this API.

**1.INTRODUCTION**

**1.1 Introduction To Indowordnet**

**WordNet** is a machine readable lexical database for English language developed at Princeton University. It has evolved as the most valuable resource for the natural language processing application. Following the Princeton WordNet, wordnets for many other languages were developed across the globe. The first wordnet for Indian languages is Hindi wordnet[3], developed at Indian Institute of Technology, Bombay. Recently, efforts are going on to develop wordnets for many Indian Languages.

**IndoWordNet**[[1]](https://en.wikipedia.org/wiki/IndoWordNet" \l "cite_note-PushpakBhattacharyya-1) is a linked lexical knowledge base of wordnets of 18 [scheduled languages of India](https://en.wikipedia.org/wiki/Scheduled_languages_of_India), viz., Assamese, Bangla, Bodo, Gujarati, Hindi, Kannada, Kashmiri, Konkani, Malayalam, Meitei (Manipuri), Marathi, Nepali, Odia, Punjabi, Sanskrit, Tamil, Telugu and Urdu.

Based on the English and Hindi WordNet we are developing Gujarati WordNet. Gujarati WordNet contains Gujarati words used in a family‘s day to day life. It groups words into sets of synonyms called synsets, provides short definitions and usage examples, and records a number of relations among these synsets or their members.

**1.2 Principles Of Wordnet construction**

* The wordnets follow the principles of minimality, coverage and replaceability for the synsets. That means, there should be at least a 'core' set of lexemes in the synset that uniquely give the concept represented by the synset (minimality), e.g., {house, family} standing for the concept of 'family' ("she is from a noble house").
* The synset should cover ALL the words representing the concept in the language (coverage), e.g., the word 'menage' will have to appear in the 'family' synset, albeit, towards the end of the synset, since its usage is rare.
* Finally, the words towards the beginning of the synset should be able to replace one another in reasonable amount of corpora (replaceability), e.g., 'house' and 'family' can replace each other in the sentence "she is from a noble house".

**1.3 Objectives and Scope Of Wordnet**

**1.3.1 Objectives**

* Get easy Accessto synset words and their examples.
* Semantic relation between words can be found out.
* Relation between different languages wordnet

**1.3.2 Scopes**

* Due to the difference in database structure of various languages compatibility in code will occur
* WordNet does not include information about the [etymology](https://en.wikipedia.org/wiki/Etymology) or the pronunciation of words and it contains only limited information about usage
* WordNet aims to cover most of everyday English and does not include much domain-specific terminology

**2.BACKGROUND THEORY**

The Hindi WordNet is a system for bringing together different lexical and semantic relations between the Hindi words. It organizes the lexical information in terms of word meanings and can be termed as a lexicon based on psycholinguistic principles. The design of the Hindi WordNet is inspired by the famous English WordNet.

In the Hindi WordNet, the words are grouped together according to their similarity of meanings. Two words that can be interchanged in a context are synonymous in that context. For each word there is a synonym set, or synset, in the Hindi WordNet, representing one lexical concept. This is done to remove ambiguity in cases where a single word has multiple meanings. Synsets are the basic building blocks of WordNet. The Hindi WordNet deals with the content words, or open class category of words. Thus, the Hindi WordNet contains the following category of words- Noun, Verb, Adjective and Adverb.

Each entry in the Hindi WordNet consists of following elements:

**1. Synset**: It is a set of synonymous words. For example, ― ͪवɮयालय, पाठशाला, èकू ल

**2**. **Gloss:** It describes the concept. It consists of two parts: Text definition: It explains the concept denoted by the synset. For example, ― वह èथान जहाँĤाथͧमक या माÚयͧमक èतर कȧ औपचाǐरक ͧश¢ा दȣ जाती है

**3. Position in Ontology:** An ontology is a hierarchical organization of concepts, more specifically, a categorization of entities and actions. For each syntactic category namely noun, verb, adjective and adverb, a separate ontological hierarchy is present.

Example of Hindi Dataset:

**Concept:** जो प्रवीष्ट न हुआ हो

**Example:** “अप्रवीष्ट महेमानो को शीघ्र ही भीतर प्रवेश करने दे”

**Synset:** अप्रवीष्ट

**3.DOMAIN KNOWLEDGE**

**3.1 GUJARATI DATASET INFORMATION**

Each entry in the Gujarati WordNet consists of following elements:

**ID** :: 1

**CAT** :: ADJECTIVE

**CONCEPT** :: જેણે જન્મ ન લીધો હોય

**EXAMPLE** :: "બ્રહ્મ અજન્મા છે."

**SYNSET-GUJARATI** :: અજન્મા, અજાત, અજ, અજન, અજન્મ, અનાગત, અયોનિ, અનુત્પન્ન

**3.2 INFLUENCE OF HINDI ON GUJARATI**

As an Indo-Aryan language, Gujarati language is very similar to Hindi. A brief comparison of Gujarati with Hindi is as follows,

* **Gender**: Gujarati language defines three genders while Hindi has only 2 genders

Gujarati:- પુલ્લિંગ, સ્ત્રીલિંગ, નપુંસક

Hindi:- **पुरुष जाति**, **स्त्री जाति**

* **Writing system:** Gujarati does not have the upper horizontal line running above the letter and few characters are modified.
* **Causative verbs:** Both Hindi and Gujarati handle causative verbs in the same fashion.

For e.g: पकवाना – Pakvaana – to cause to cook

मैंने आज बहन से खाना बनवाया ।

Maine aaj behan se khaana banvaaya

Today i made my sister to cook.

* **‘Want’ and ‘should’:** Both Hindi and Gujarati handle “I should ...” and “I want ..” in a similar ways. Gujarati uses ‘jo’ which is similar to ‘chah’ of Hindi.

For e.g. ‘ I should go home now.’ is written as,

Hindi, ‘मुजे घर जाना चाहीये।’

Gujarati, ‘મારે ઘરે જવુ જોઇએ.’

As Gujarati language is closely related to Hindi, the most of Gujarati synsets are created by translating Hindi synsets to Gujarati synsets. However, emphasis was given to understand the concept independently of a language and then to create synset. Though notion of concept is defined independently of the language, many times it was observed that the concept present in Hindi was not present in Gujarati or even though the concept was present there was no indigenous lexeme for the concept

**3.3 ISSUES RELATED TO SYNSET DEVELOPMENT**

During the development of synsets, some disagreements were observed between Hindi concepts and Gujarati concepts.

**3.3.1**. **Hindi synsets not linked with Gujarati**

Following are some examples of Hindi synsets not linked with Gujarati,

• Difference in concept description

**Concept**: तुरही की तरह का एक बड़ा बाजा

**Example:** “नरिसहा की आवाज़ दूर-दूर तक सुनाई देती है”

**Synset:** नरिसहा, नरिसगा, गोमुख

No such concept is identified in Gujarati language. However, there is a concept in Gujarati language for similar instrument which is used at war-front to announce beginning of a war.

• No indigenous lexeme in Gujarati

**Concept**: इत्र का व्यापार करनेवाला व्यक्ती

**Example:** “आजकल, इत्र व्यापारी नक़ली इत्र का व्यापार भी करने लगे हैं”

**Synset:** इत्र व्यापारी, इत्र फरोश, इत्र फ़रोश, अत्तार, गंधी, गन्धी

There is no indigenous lexeme for this concept in Gujarati language.

• Confusing gloss

**Concept:** एक छोटा पक्षी जो प्रायः अपना घोसला मकानो में बनाता है

**Example:**“गौरैया अपने बच्चो को दाना चुगा रही ह”

**Synset:** गौरैया, गौरेया, वृषायण, आकली

The concept is general and exists in Gujarati language but it is difficult to identify the Gujarati name of the bird from the synset.

• Difficult to adopt

**Concept:** जो प्रवीष्ट न हुआ हो

**Example:** “अप्रवीष्ट महेमानो को शीघ्र ही भीतर प्रवेश करने दे”

**Synset:** अप्रवीष्ट

Though this word can be translated in Gujarati language, it is not a native concept used in Gujarati language.

• No such concept in Gujarati

**Concept:** जो अकेला चरता या वीचरण करता हो

**Example:** “जंगली सूअर एक पृथकचर पशु है”

**Synset:** पृथकचर

There is no such concept in Gujarati language

**3.3.2 Language specific synset**

While major part of the day to day vocabulary of Gujarati language is similar to that of Hindi, there are some concepts which are very specific to Gujarati language. These concepts are very specific to the culture of Gujarat. These concepts refer to food items, places, traditions, religion etc. Some of the examples are as follows:

• Culture specific concept

**Concept :** કોઇ ખાસ પ્રસંગે કસુંબો પીવા માટે ભેગા થવું

**Example :** “ગુજરાત ના કોઇ ગામો માં આજે પણ ડાયરા થાય છે”

**Synset :** ડાયરો (डायरो, Daayaro)

• Tradition specific concept

**Concept :** એક ફળ કે જે લગ્ન પ્રસંગે વર કન્યા ના હાથે બાંધે છે

**Example :** “લગ્ન પછી વર કન્યા મીંઢળ છોડે છે”

**Synset :** મીંઢળ (मीढळ, mIMdhaL)

• religion specific concept

**Concept :** મોક્ષ માટે ભગવાન નું નામ લેતા લેતા ગીરનાર પર થી પડતું મુકવું.

**Example :** “ગીરનાર શીખર પર થી ભક્તો ભૈરવજપ કરતા હતા”

**Synset** : ભૈરવજપ (भैरवजप, bheiravajapa)

**4.RELATED WORK**

**4.1 TERMINOLOGIES OF WORDNET**

* **SYNSETS**

Words from the same lexical category that are roughly synonymous are grouped into [synsets](https://en.wikipedia.org/wiki/Synsets). Synsets include simplex words as well as [collocations](https://en.wikipedia.org/wiki/Collocation) like "eat out" and "car pool."

All synsets are connected to other synsets by means of semantic relations. These relations, which are not all shared by all lexical categories, include:

* [Nouns](https://en.wikipedia.org/wiki/Noun)
  + [*hypernyms*](https://en.wikipedia.org/wiki/Hypernym): *Y* is a hypernym of *X* if every *X* is a (kind of) *Y* (*canine* is a hypernym of [*dog*](https://en.wikipedia.org/wiki/Dog))
  + [*hyponyms*](https://en.wikipedia.org/wiki/Hyponym): *Y* is a hyponym of *X* if every *Y* is a (kind of) *X* (*dog* is a hyponym of *canine*)
  + *coordinate terms*: *Y* is a coordinate term of *X* if *X* and *Y* share a hypernym (*wolf* is a coordinate term of *dog*, and *dog* is a coordinate term of *wolf*)
  + [*meronym*](https://en.wikipedia.org/wiki/Meronymy): *Y* is a meronym of *X* if *Y* is a part of *X* (*window* is a meronym of *building*)
  + [*holonym*](https://en.wikipedia.org/wiki/Holonymy): *Y* is a holonym of *X* if *X* is a part of *Y* (*building* is a holonym of *window*)
* [Verbs](https://en.wikipedia.org/wiki/Verb)
  + *hypernym*: the verb *Y* is a hypernym of the verb *X* if the activity *X* is a (kind of) *Y* (*to perceive* is an hypernym of *to listen*)
  + [*troponym*](https://en.wikipedia.org/wiki/Troponym): the verb *Y* is a troponym of the verb *X* if the activity *Y* is doing *X* in some manner (*to lisp* is a troponym of *to talk*)
  + [*entailment*](https://en.wikipedia.org/wiki/Entailment): the verb *Y* is entailed by *X* if by doing *X* you must be doing *Y* (*to sleep* is entailed by *to snore*)
  + *coordinate terms*: those verbs sharing a common hypernym (*to lisp* and *to yell*)
* **LEMMAS**

A lemma is wordnet's version of an entry in a dictionary: A word in canonical form, with a single meaning. E.g., if you wanted to look up "banks" in the dictionary, the canonical form would be "bank" and there would be separate lemmas for the nouns meaning "financial institution" and "side of the river", a separate one for the verb "to bank (on)", etc.

**4.2 WORDNET::SIMILARITY**

* WordNet::Similarity is a freely available software package that makes it possible to measure the semantic similarity and relatedness between a pair of concepts (or synsets).
* It provides six measures of similarity, and three measures of relatedness, all of which are based on the lexical database WordNet.
* These measures are implemented as Perl modules which take as input two concepts, and return a numeric value that represents the degree to which they are similar or related.

**4.2.1 Similarity are Groups based on Parameter such as**

**1.** **metrics belong to the thesaurus-based ones:**

* Path similarity
* Leacock-Chodorow Similarity (Leacock and Chodorow 1998)
* Wu-Palmer Similarity (Wu and Palmer 1994)

**2.** **metrics belong to the thesaurus- and corpus-based ones (also called the Information Content metrics):**

* Resnik Similarity (Resnik 1995)
* Lin Similarity (Lin 1998b)
* Jiang-Conrath distance (Jiang and Conrath 1997**)**

**4.2.2 Path Similarity**

Path similarity computes shortest number of edges from one word sense to another word sense, assuming a hierarchical structure like WordNet (essentially a graph). In general, word senses which have a longer path distance are less similar than those with a very short path distance

* **Problem with basic path-based similarity**
* Pathsimilarity is not commutative
* Assumes each link represents a uniform distance
* But, some areas of WordNet are more developed than others
* Depended on the people who created it
* Also, links deep in the hierarchy are intuitively more narrow than links higher up [on slide 4, e.g., nickel to money vs nickel to standard]

**For example:**

similarity\_path(synsets('પીસવું')[2],synsets('પીસવું')[1])

0.2

**4.2.3 Wu-Palmer Similarity (Wu and Palmer 1994)**

The Wu & Palmer calculates relatedness by considering the depths of the two synsets in the WordNet taxonomies, along with the depth of the LCS (Least Common Subsumer).

**Wup = 2 \* depth (lcs) / (depth (s1) + depth (s2))**.

This means that 0 < score <= 1. The score can never be zero because the depth of the LCS is never zero (the depth of the root of a taxonomy is one). The score is one if the two input concepts are the same.

**For example**: similarity\_wup(synsets('પીસવું')[2],synsets('પીસવું')[1])

0.25

Thedisadvantage being an inherent disadvantage, in which two concepts in the same hierarchy may show a lower similarity than two concepts belonging to different hierarchies.

The following modification is proposed to remedy the disadvantage mentioned above

N=depth(lch) N1 =depth(s1) N2 =depth(s2)

**wup(s1,s2) = 2 N/(N1+N2) if N1 != N and N2!=N**

**2N/(N2-N) if N1 = N**

**2N/(N1-N) if N2 = N**

1.Two concepts belong to different hierarchies if N1 != N and N2!=N

2. Two concepts belong to same hierarchy if **N1 = N or N2 = N**

**4.2.4 Leacock-Chodorow Similarity (Leacock and Chodorow 1998)**

* Ba**s**ed on simple edge counts in the is a hierarchy of wordnet
* Deals with nouns only
* The path length is scaled by the depth of the taxonomy

LCS(c1,c2) = -log(path\_length/2 \*D)

where c1,c2 are the concepts and D is the depth of taxonomy

* + Depth shows specificity but not frequency . Low frequency concepts often much more specific than high frequency ones

**For Example:** similarity\_lch(synsets('ઇલા')[0],synsets('ઇલા')[1])

**0.6532**

**5.SOFTWARE REQUIREMENTS**

**(Tools And Technologies Used)**

***5.1 Python***

Python’s strengths are in its text, list, and structure support. Structures are weakly typed, but supported by a powerful set of language constructs in the form of list comprehensions and lambda functions. Also, as a modern general purpose scripting language, Python is ideal for prototyping. This makes it ideal for quick development – even if final processing times might be slow.

 Python has the open source Natural Language Toolkit (NLTK). This library is a rich set of natural language processing tools and datasets, intended for educational purposes. As such, the combination of NLTK and Python make the perfect combination for teaching NLP techniques, because powerful examples can be demonstrated with short sections of code. Students can then quickly modify this code in a meaningful way.

***5.2 NLTK***

The Natural Language Toolkit, or more commonly **NLTK**, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language.

The **NLTK corpus** is a massive dump of all kinds of natural language data sets that are definitely worth taking a look at. Almost all of the files in the **NLTK corpus** follow the same rules for accessing them by using the **NLTK** module, but nothing is magical about them

Install several free software packages.Current download pointers and instructions are available at *http://www.nltk.org/*.

**5.3 Jupyter Notebook**

.

Jupyter Notebooks are a powerful way to write and iterate on your Python code for data analysis. Rather than writing and re-writing an entire program, you can write lines of code and run them one at a time.

Jupyter Notebook is built off of IPython, an interactive way of running Python code in the terminal using the REPL model (Read-Eval-Print-Loop). The IPython Kernel runs the computations and communicates with the Jupyter Notebook front-end interface. It also allows Jupyter Notebook to support multiple languages. Jupyter Notebooks extend IPython through additional features, like storing your code and output and allowing you to keep markdown notes.

**5.4 Packages Required**

**5.4.1 Pickle Module**

* The pickle module is used for implementing binary protocols for serializing and de-serializing a Python object structure.
* Pickling: It is a process where a Python object hierarchy is converted into a byte stream.
* Unpickling: It is the inverse of Pickling process where a byte stream is converted into an object hierarchy.
* Some of the methods used of pickle Module while implementing wordnet are :

(1) **pickle.dumps(obj, protocol = None, \*, fix\_imports = True)**

This function returns the pickled representation of the object as a bytes object.

**(2)pickle.load(file, \*, fix\_imports = True, encoding = “ASCII”, errors = “strict”)**  
 This function is equivalent to Unpickler(file).load(). This function is used to read a pickled object representation from the open file object file and return the reconstituted object hierarchy specified.

**5.4.2 CSV Module**

The so-called CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases. There is no “CSV standard”, so the format is operationally defined by the many applications which read and write it.

The **[csv](https://docs.python.org/2/library/csv.html" \l "module-csv" \o "csv: Write and read tabular data to and from delimited files.)** module defines the following functions:

(1) csv.**reader()**

Return a reader object which will iterate over lines in the given csvfile. csvfile can be any object which supports the [iterator](https://docs.python.org/2/glossary.html#term-iterator)protocol and returns a string each time its **next()** method is called — file objects and list objects are both suitable.

**6.IMPLEMENTATION DETAILS**

## 6.1 Installing NLTK through Anaconda

**Step1)** Please install anaconda (which can also be used to install different packages) by visiting <https://www.anaconda.com/download/> and select which version of python you need to install for anaconda.

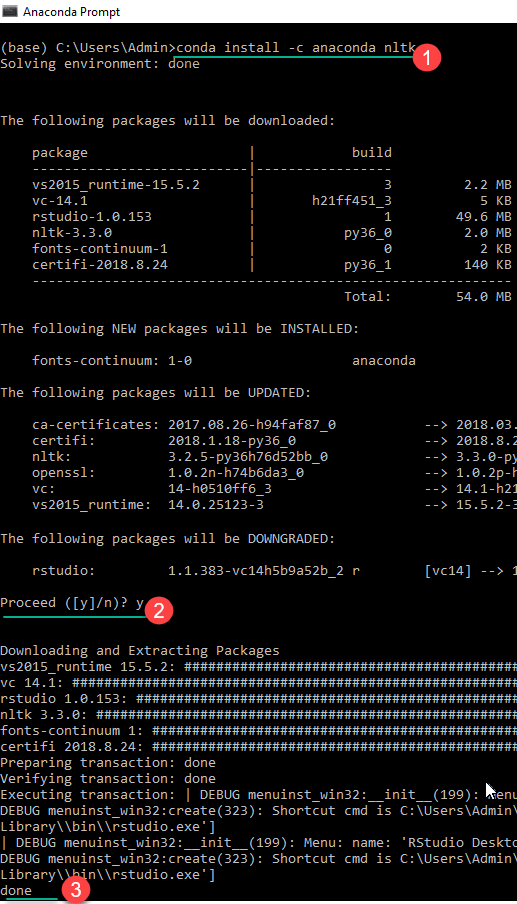
**Step 2)**In the Anaconda prompt,

1. Enter command

conda install -c anaconda nltk

2. Review the package upgrade, downgrade, install information and enter yes

3.NLTK is downloaded and installed



**6.2 How to Download all packages of NLTK**

**Step 1)**Run the Python interpreter in Windows or Linux

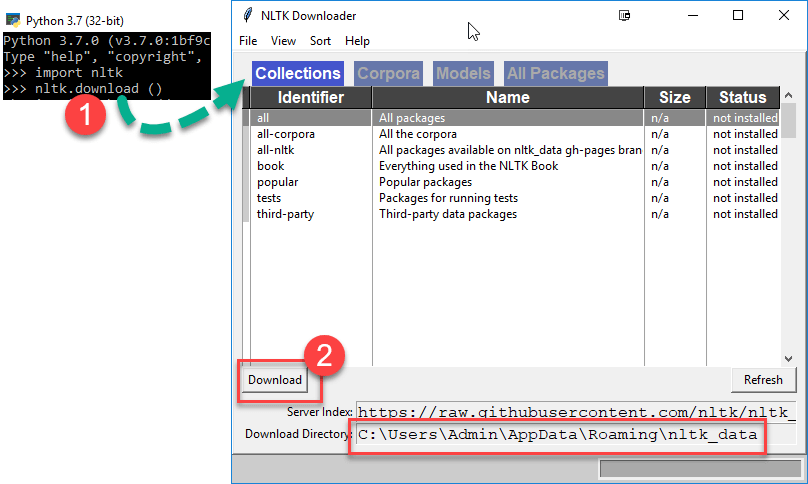
**Step 2)**

1. Enter the commands

import nltk

nltk.download ()

1. NLTK Downloaded Window Opens. Click the Download Button to download the dataset. This process will take time, based on your internet connection



**Step 3)**To test the installed data use the following code

