pyiwn: A Python-based API to access Indian Language WordNets

Ritesh Panjwani†, **Diptesh Kanojia**†,♣,⋆, **Pushpak Bhattacharyya**†
†Indian Institute of Technology Bombay, India
♣IITB-Monash Research Academy, India
⋆Monash University, Australia
†{riteshp, diptesh, pb}@cse.iitb.ac.in

Abstract

Indian language WordNets have their individual web-based browsing interfaces along with a common interface for IndoWordNet. These interfaces prove to be useful for language learners and in an educational domain, however, they do not provide the functionality of connecting to them and browsing their data through a lucid application programming interface or an API. In this paper, we present our work on creating such an easy-to-use framework which is bundled with the data for Indian language WordNets and provides NLTK WordNet interface like core functionalities in Python. Additionally, we use a pre-built speech synthesis system for Hindi language and augment Hindi data with audios for words, glosses, and example sentences. We provide a detailed usage of our API and explain the functions for ease of the user. Also, we package the IndoWordNet data along with the source code and provide it openly for the purpose of research. We aim to provide all our work as an open source framework for further development.

1 Introduction

WordNets are extensively used in many sub-tasks for Natural language Processing (NLP) (Knight and Luk, 1994; TufiŞ et al., 2004). They are a rich semantic lexicon which are accessible, free-to-use and fairly accurate. They have been used in cross-lingual information retrieval (Gonzalo et al., 1998), word sense disambiguation (Sinha et al., 2006), question answering (Pasca and Harabagiu, 2001) etc. Princeton WordNet (Fellbaum, 1998) or the English WordNet was the first to come into existence. EuroWordNet (Vossen, 1998) followed with a common structure for 12 European languages. Indian language WordNets originated with the advent of Hindi WordNet (Narayan et al., 2002) and based on an expansion approach, the rest of them were created. They form a common lexico-semantic resource called IndoWordNet (IWN) (Bhattacharyya, 2010). India has more than 22 languages and 18 of these have constituent WordNets under a common roof - IndoWordNet. A considerable effort has gone into the creation of a perfect Application Programming Interface (API) for English WordNet and many of these are available for use, publically. Although, we do not see that kind of push for a common API for Indian language WordNets. NLP research for Indian languages has seen tremendous growth in the recent past and wordnets are a crucial resource especially in the context of NLP methodologies based on knowledge bases.

"With our work, we aim to provide an accessible, robust, easy-to-use API for Indian language WordNets."

Additionally, this will help emerging wordnets acquire our open-source framework and adapt to it for creating a simple python based API, thus helping NLP for their own language.

2 Motivation

Efforts to create a lexical semantic network for Indian languages began with Hindi WordNet[1] (Narayan et al., 2002), and based on the concept of pivotal expansion, IndoWordNet[2] (Bhattacharyya, 2010) was created. NLP for Indian languages is gaining traction among the computer scientists in India, and a robust framework which is readily available for use is much needed. We believe that such an API bundled with the IndoWordNet data could be really helpful to the NLP community.

[1]`http://www.cfilt.iitb.ac.in/wordnet/webhwn/index.php`
[2]`http://www.cfilt.iitb.ac.in/indowordnet/`

Princeton WordNet or the English WordNet API is available for use via NLTK[3] (Bird et al., 2009) in Python. Bond et al. (2016) collaborate many WordNets and provide open access for using the wordnet data aligned with them. These wordnets from all over the world are linked via their linkages to English WordNet. Indian language wordnets are also linked to English, but only 25000 out of 40000+ synsets. Until the time all the synsets are unlinked; a separate API for browsing through Indian languages is required, and a common API might not sufficiently cover the data available with IndoWordNet.

Hence, we build this API with an aim that IndoWordNet data should also readily available in an easy-to-use framework. Python facilitates prebuilt libraries and datasets for NLP via NLTK. TensorFlow by Google (Abadi et al., 2016) is also built on Python, and other classic Machine Learning algorithms are available for use via the *sci-kit* learn (*sklearn*) library (Pedregosa et al., 2011). Hence, we choose Python for implementing the API and build a framework using it.

3 Related Work

The Java WordNet Library[4] has been extensively used for research across various domains in NLP (Chauhan et al., 2013; Zesch et al., 2008; Gurevych et al., 2012). extJWNL[5] extend JWNL and provides command-line support, and Maven[6] support among many other features in their API. Emerging WordNets like Sinhala WordNet (Welgama et al., 2011) employ JWNL to create an API for their WordNet. Java API for WordNet Searching (JAWS) (Spell, 2009) is another such implementation. The MIT Java WordNet Interface (JWI)[7] is also available for the same purposes and is available under the Creative Commons 4.0 License[8]. Finlayson (2014) presents an extensive evaluation of the APIs available in Java for accessing Princeton WordNet. All of the work above has been done for Java, and is available for Princeton WordNet. A Python based toolkit, ESTNLTK (Orasmaa et al., 2016) includes Estonian WordNet developed under the EuroWordNet project (Vossen, 1998).

Previously, efforts had been made to create an API for IndoWordNet but they are not Python based. Prabhugaonkar et al. (2012) describe a two-layered architecture of a web-based API created using PHP. It requires one to download data separately and is inconvenient to deploy; we also come across hard-coded paths while trying to deploy their API. A Java-based API[9] is available for download on the Hindi WordNet web interface, and also requires one to separately download the database for Hindi WordNet. Redkar et al. (2016) claim to have built an API for WordNets universally but their work is not publicly accessible, and no references to their implementation could be found. Hence, we work on an API which would contain NLTK like functionality, should be robust, readily available, and more importantly easy-to-use.

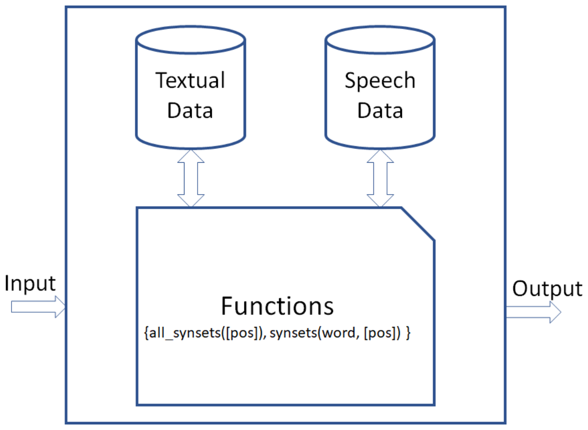

Figure 1: Basic flow of the *pyiwn* API

4 API Design

We choose Python for implementation due to its widespread use in the NLP community and aim to align our work with NLTK. With this in mind, we keep the design of our API similar to that of NLTK WordNet Interface[10]. Our API provides access to synsets and their relational connections such as hypernymy, hyponymy, meronymy, *etc.* with other synsets for all the languages mentioned in the Table 1. The basic flow of the API is visualized in Figure 1. The API is available as an open source project on GitHub[11].

[3]http://www.nltk.org/

[4]http://jwordnet.sourceforge.net/handbook.html

[5]http://extjwnl.sourceforge.net/

[6]https://maven.apache.org/

[7]https://projects.csail.mit.edu/jwi/

[8]https://creativecommons.org/licenses/by/4.0/

[9]https://goo.gl/N8GXAU

[10]http://www.nltk.org/howto/wordnet.html

[11]https://github.com/riteshpanjwani/pyiwn

Language	Noun	Verb	Adjective	Adverb	Total
Hindi	29807	3687	6336	541	40371
Assamese	9065	1676	3805	412	14958
Bengali	27281	2804	5815	445	36346
Bodo	8788	2296	4287	414	15785
Gujarati	26503	2805	5828	445	35599
Kannada	12765	3119	5988	170	22042
Kashmiri	21041	2660	5365	400	29469
Konkani	23144	3000	5744	482	32370
Malayalam	20071	3311	6257	501	30140
Manipuri	10156	2021	3806	332	16351
Marathi	23271	3146	5269	539	32226
Nepali	6748	1477	3227	261	11713
Odiya	27216	2418	5273	377	35284
Punjabi	23255	2836	5830	443	32364
Sanskrit	32385	1246	4006	265	37907
Tamil	16312	2803	5827	477	25419
Telugu	12078	2795	5776	442	21091
Urdu	22990	2801	5786	443	34280

Table 1: Statistics of the synsets in IndoWordNet

4.1 Data

The data for all the 18 languages is stored in the file system. The organization of the data is described as follows:

- **Synsets**. All the synset data for each language in IndoWordNet is stored in a different file. In each file, on each line, there is a synset with its unique identifier, the synset, gloss, examples, and the part of speech of the synset. There are 4 more files for each language for each of the part of speech: noun, verb, adjective, and adverb. These files contain the synsets for the respective part of speech of the synset.

- **Words**. This type of file contains all the unique words available in the WordNet along with its synset unique identifier and part of speech tag. Similar to Synsets file, there is separate file for each language. Each such file contains all the words in the WordNet of the respective language. There are 4 more files for each language that has words for the respective part of speech of the synset.

- **Synset relations**. This type of file stores various lexico-semantic relations among the synsets. Since, the Indian language WordNets are based on Hindi WordNet, all the relations in Hindi WordNet are also valid for other language WordNets in the IndoWordNet.

- **Ontology nodes**. The next is ontology nodes file which contains a list of nodes that the synset belongs to in an ontology tree (Fensel, 2001).

Apart from the textual data, we also specifically augment Hindi WordNet with speech data for all the words, glosses and example sentences. The speech data is in Waveform Audio File Format (WAV). This data is obtained using Indic TTS, a text-to-speech synthesis system for Indian languages (Patil et al., 2013).

Our system provides access to IWN data for the languages mentioned in Table 1. The number of synsets present in our database are also present in the table above. Figure 1 shows the architecture for our system.

4.2 Features

Our API provides access to the synset data and its lexico-semantic relations with other synsets for all the languages in IndoWordNet. The API module can imported in the following manner:

```
>>> from pyiwn import pyiwn
>>> iwn = pyiwn.IndoWordNet(lang)
```

The class *IndoWordNet* in the *pyiwn* module takes language (*lang*) as an argument. In this way, an object *iwn* is created to access the synset and speech data from WordNet of that language. The core features provided are described in the following sections.

4.2.1 Synsets

The API returns a *Synset* object for all the functions that are described ahead. The *Synset* object holds the information described in Section 4.1. The synsets can be accessed using in the following ways:

Access to all synsets

```
>>> iwn.all_synsets()
```

This function gives access to all the synsets for the given language.

```
>>> iwn.all_synsets(pos=pos_tag)
```

The above line signifies that the API will give all the synsets for a given language where the *pos_tag* can hold a string value from {noun, verb, adjective, adverb}.

Access to synsets of a given word

```
>>> iwn.synsets(word)
```

This functions searches for all the synsets that contain the given *word* and returns a Python *list* of *Synset* objects that have all the properties of a synset described the next section.

```
>>> iwn.synsets(word, pos=pos_tag)
```

Similarly, this function is used to filter the results for a given *word* by an optional second argument, *pos_tag*.

4.2.2 Synset properties

The synset has the properties like, head word (first word of the synset), POS tag, gloss (definition of the synset), examples, lemma names, ontology nodes and relations which is described in detail in Section 4.1. The API has a *Synset* class that has all of these mentioned properties as functions. The below code examples demonstrate the functions of the *Synset* class.

# creates a list of *Synset* objects for the given *word* and returns the first *Synset* object
```
>>> syn = iwn.synsets(word)[0]
```

# returns part of speech tag
```
>>> syn.pos()
```

# returns head word
```
>>> syn.head_word()
```

# returns definition
```
>>> syn.gloss()
```

# returns a list of examples
```
>>> syn.examples()
```

# returns a list of ontology nodes
```
>>> syn.ontology_nodes()
```

# returns a list of lemmas
```
>>> syn.lemma_names()
```

# returns a dictionary of relations
```
>>> syn.relations()
```

4.2.3 Words

The API also provides functions to access only words of a particular language. The below code examples show the usage of this functionality.

# returns a list of all the words in the given language
```
>>> iwn.all_words()
```

# returns a list of all the words in the given language filtered by given an optional argument, *pos_tag*
```
>>> iwn.all_words(pos=pos_tag)
```

4.2.4 Speech

The speech data for Hindi words can also be accessed via the API using the following function that takes a *word* as an argument and returns the WAV file object.[12]

# returns a WAV file object for a given *word*
```
>>> iwn.word_speech(word)
```

# For the speech of the glosses and examples, first create a list of *Synset* objects for the given *word* and returns the first *Synset* object
```
>>> syn = iwn.synsets(word)[0]
```

# returns a WAV file object for a given *gloss*
```
>>> syn.gloss_speech(gloss)
```

# returns a list of WAV file objects for a given list of *examples*
```
>>> syn.examples_speech(examples)
```

4.2.5 Morphological Analyzer

The role of morphological analyzers is to find the dictionary form of the word by restoring the changes caused by inflectional or derivational morphology of the language (*nationalism* $\rightarrow$ *nation*) (Buckwalter, 2002).

The API provides a function that takes in a word of a given language and returns the dictionary form of the word (lemma) which can then be passed on to other functions in the API.

# returns a lemma for the given *word*
```
>>> iwn.morph(word)
```

This functionality enables us to find the related synsets for many morphological variants of the

[12]https://docs.python.org/2/library/wave.html

words. This is really helpful in case of morphologically rich languages like Marathi. Currently, this feature is available only for the languages Hindi and Marathi. We plan to include the morphological analyzer for other languages in the future.

5 Conclusion & Future work

We provide an API for accessing IndoWordNet using Python. WordNet package in NLTK is already widely used amongst NLP researchers; we provide them with a similar functionality for Indian language WordNets and bundle the data along with. We also provide audio data in a separate package for the Hindi language. Currently, we only provide with functionalities such as displaying synset data, browsing through relational connections of a synset with other synsets, and access to speech data for the Hindi language. We believe our work will help the NLP community by providing them with a robust and easy-to-use framework for Indian languages.

In future, we plan to add functionalities like getting the top-level relational synset, the path-length of longest and shortest relational synset, finding all parent/child synsets with respect to a lexico-semantic relation. We also plan to create voice models using a speech synthesis system for other Indian languages or use pre-built voice models to generate audios for Indian languages and augment our API with the audio data. In addition to this, we aim to provide morphological analysis for more languages other than Hindi and Marathi as a feature for the ability to search concepts using the inflectional form of a word. We hope our work helps the Indian NLP diaspora further their research and gain more insights.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

P Bhattacharyya. 2010. Indowordnet. lexical resources engineering conference 2010 (lrec 2010). *Malta, May*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Francis Bond, Piek Vossen, John P McCrae, and Christiane Fellbaum. 2016. Cili: the collaborative interlingual index. In *Proceedings of the Global WordNet Conference*, volume 2016.

Tim Buckwalter. 2002. Buckwalter {Arabic} morphological analyzer version 1.0.

Rashmi Chauhan, Rayan Goudar, Robin Sharma, and Atul Chauhan. 2013. Domain ontology based semantic search for efficient information retrieval through automatic query expansion. In *Intelligent Systems and Signal Processing (ISSP), 2013 International Conference on*, pages 397–402. IEEE.

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.

Dieter Fensel. 2001. Ontologies. In *Ontologies*, pages 11–18. Springer.

Mark Alan Finlayson. 2014. Java libraries for accessing the princeton wordnet: Comparison and evaluation. In *Proceedings of the 7th Global Wordnet Conference, Tartu, Estonia*, volume 137.

Julio Gonzalo, Felisa Verdejo, Irina Chugur, and Juan Cigarran. 1998. Indexing with wordnet synsets can improve text retrieval. *arXiv preprint cmp-lg/9808002*.

Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M Meyer, and Christian Wirth. 2012. Uby: A large-scale unified lexical-semantic resource based on lmf. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 580–590. Association for Computational Linguistics.

Kevin Knight and Steve K Luk. 1994. Building a large-scale knowledge base for machine translation. In *AAAI*, volume 94, pages 773–778.

Dipak Narayan, Debasri Chakrabarti, Prabhakar Pande, and Pushpak Bhattacharyya. 2002. An experience in building the indo wordnet-a wordnet for hindi. In *First International Conference on Global WordNet, Mysore, India*.

Siim Orasmaa, Timo Petmanson, Alexander Tkachenko, Sven Laur, and Heiki-Jaan Kaalep. 2016. Estnltk - nlp toolkit for estonian. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may. European Language Resources Association (ELRA).

Marius Pasca and Sanda Harabagiu. 2001. The informative role of wordnet in open-domain question answering. In *Proceedings of NAACL-01 Workshop on WordNet and Other Lexical Resources*, pages 138–143.

Hemant A Patil, Tanvina B Patel, Nirmesh J Shah, Hardik B Sailor, Raghava Krishnan, GR Kasthuri, T Nagarajan, Lilly Christina, Naresh Kumar, Veera Raghavendra, et al. 2013. A syllable-based framework for unit selection synthesis in 13 indian languages. In *Oriental COCOSDA held jointly with 2013 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE), 2013 International Conference*, pages 1–8. IEEE.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830.

Neha R Prabhugaonkar, A Nagvenkar, and R Karmali. 2012. Indowordnet application programming interfaces.

Hanumant Redkar, Sudha Bhingardive, Kevin Patel, Pushpak Bhattacharyya, Neha Prabhugaonkar, Apurva Nagvenkar, and Ramdas Karmali. 2016. Wwds apis: application programming interfaces for efficient manipulation of world wordnet database structure. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Manish Sinha, Mahesh Reddy, and Pushpak Bhattacharyya. 2006. An approach towards construction and application of multilingual indo-wordnet. In *3rd Global Wordnet Conference (GWC 06), Jeju Island, Korea*.

Brett Spell. 2009. Java api for wordnet searching (jaws). *URL http://lyle. smu. edu/˜ tspell/jaws*.

Dan TufiŞ, Radu Ion, and Nancy Ide. 2004. Fine-grained word sense disambiguation based on parallel corpora, word alignment, word clustering and aligned wordnets. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1312. Association for Computational Linguistics.

Piek Vossen. 1998. *A multilingual database with lexical semantic networks*. Springer.

Viraj Welgama, Dulip Lakmal Herath, Chamila Liyanage, Namal Udalamatta, Ruvan Weerasinghe, and Tissa Jayawardana. 2011. Towards a sinhala wordnet. In *Proceedings of the Conference on Human Language Technology for Development*.

Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Using wiktionary for computing semantic relatedness. In *AAAI*, volume 8, pages 861–866.