TECHNOLOGY REVIEW

Gensim: A Toolkit for Modern Research

Harita Reddy

University of Illinois at Urbana-Champaign

ARTICLE HISTORY
Compiled November 7, 2021

1. Introduction

Vector Space Modeling (VSM) is one of the most important topics in Text Information Systems and has a number of applications, ranging from retrieving similar documents to identifying fake news based on the text. Gensim is a toolkit that has made the generation and training of word vectors very convenient for users with different levels for experience in Python[1]. In addition to VSM, Gensim supports other tasks related to Text Information Systems, including Topic Modeling. In this paper, I give an overview of Gensim as a Python package, discuss some related packages, and finally discuss three major applications of the package.

2. Overview of Python Package

Gensim is available as a Python package for Information Retrieval and Natural Language Processing tasks and supports a variety of applications[1].

The most important task that Gensim supports is Vector Space Modeling (VSM). Two of the most popular vector models supported by Gensim are *Word2Vec* and *FastText*. Word2Vec[6] is a methodology to generate word vectors by leveraging neural networks and most popular neural network architectures used for this are Continuous Bag of Words (CBOW) and Skipgram. FastText[7] is another alternative model introduced by Facebook and leverages n-grams to represent rarer words in a better way as compared to Word2Vec. GloVe[5] is another option available and is an unsupervised method that relies on the co-occurrences of words in a given corpus to generate word vectors.

Another important application supported by Gensim is topic modeling. The aim of topic modeling is to obtain implicit topics discussed within the text and Gensim supports Latent Dirichlet Allocation (LDA)[8] to estimate the model based on the training data and then determine the topic distribution on new documents.

Apart from the different modeling and training techniques, Gensim also supports granular steps of Natural Language Processing and Text Retrieval, including Stemming. Stemming is a pre-processing step required for many Text Retrieval applications and one of the most popular stemming algorithms, *Porter's Stemmer*[9] is directly supported by Gensim. Gensim also supports pre-processing of strings, including steps like

[1]https://pypi.org/project/gensim/

removing punctuation characters. Overall, the package is a comprehensive source of different steps within Natural Language Processing and Text Retrieval tasks.

An important feature of Gensim is that it enables access to pre-trained models and corpora, which helps users to leverage the already existing datasets. This is enabled by the *gensim-data* project[2], which is a repository of datasets and models for text processing. This repository of datasets can be directly used through the Gensim package in Python. Examples of available datasets include the *20-newsgroups* dataset[3], which is a collection of 20,000 posts in 20 Usenet newsgroups, and the *SemEval Task 3*[4] dataset, that is often used for building language models. Popular pre-trained models include *glove-twitter-200*[5], which is a vector model trained on 2 billion tweets based on GloVe. For users to add new datasets and pre-trained models, the users can create an issue and link the appropriate documentation and explanation to get the application reviewed.

One of the most interesting features of Gensim is that it supports Distributed Computing[15]. Given that many applications require to process a large number of documents (for eg., a search engine) that cannot be done on an individual machine, the concept of distributed computing allows the task to be split across multiple machines. Gensim leverages the Pyro library[4] to enable the distributed machines to talk to each other. The most important Natural Language Processing techniques that are supported by Gensim in a distributed environment are Latent Semantic Analysis and Dirichlet Allocation.

2.1. Related Packages

Gensim is dependant on the *NumPy*[21] and the *SciPy*[20] packages, which are widely used for mathematical computing. Gensim can also be used in conjunction with other Python packages. *SpaCy*[14] is a popular Natural Language Processing package in Python and provides pre-trained models as well. For topic-modeling use cases, SpaCy can be used for the pre-processing steps, including steps such as tokenization and lemmatization. Finally, Gensim's LDA model can be leveraged for extracting topics. While both Gensim and SpaCy have similar features, SpaCy is considered as more of a high-speed comprehensive Natural Language Processing library, whereas Gensim is considered as a package for modeling topics and document similarity and retrieval[5].

Another related Python package is *scikit-learn*[16]. Scikit-learn contains the LDA model and it has been noted that when Scikit-learn LDA is used with Cython, Scikit-learn is almost 3 times faster than the LDA model in Gensim.[6]. Gensim also provides scikit-learn API in the form of *gensim.sklearn_api*, using which the users can use both packages together.

---

[2]https://github.com/RaRe-Technologies/gensim-data
[3]http://qwone.com/ jason/20Newsgroups/
[4]https://pypi.org/project/Pyro4/
[5]https://stackshare.io/stackups/gensim-vs-spacy
[6]https://github.com/RaRe-Technologies/gensim/issues/457

3. Gensim Applications

3.1. Topic Modeling

Latent Dirichlet Allocation (LDA) is a popular technique for modeling topics in text. Apart from the single-core implementation in the form of *LdaModel*, Gensim supports Multicore LDA through the *ldamulticore* class, which uses multiple CPUs to reduce the training time of the models[17]. The official documentation of Gensim has specified the comparison of the wall-clock performance of single-core vs multi-core LDA and it shows that a multi-core LDA with 3 workers reduces the processing time by close to 3.4 times in the English Wikipedia corpus[7]. Topic modeling with Gensim has been used in multiple research works, including the work on analysis of Brexit Impact by Ilyas et al., where they extracted the most discussed topics on *Twitter*[18].

3.2. Text Summarization

The task of extracting the most important sentences from the text to provide a summary to the readers has some interesting applications. Gensim supports English-language extractive text summarization through a summarizer implementation based on the *TextRank* model[10], which is a Graph-based model proposed by Mihalcea and Tarau. A limitation of the summarizer implementation is that it does not support parallel processing[11]. Gensim has been widely used for text summarization research, even for languages other than English[2]. For example, Alami et al. leveraged the Gensim Word2Vec Skipgram model to build the vectors for Arabic language for Automatic Text Summarization (ATS) of emails [12].

3.3. Sentiment Analysis

Word2Vec has been commonly used for automatically getting textual features for sentiment analysis, including in the work by Haixia, where Word2vec was leveraged for analysing sentiment of citations and was found to be differentiating positive and negative sentiments among the citations[23]. Another important tool provided in Gensim is *Doc2Vec*[22]. Maslova and Potapov leveraged Gensim's Doc2Vec for sentiment analysis of short texts derived from social networking platforms[19]. Doc2Vec is somewhat similar to Word2Vec but generates embeddings for the entire documents[24], and was created because of one weakness of word embeddings - the loss of ordering and semantic information of words.

4. Support and Open Issues

Gensim is an open-source project and is always being improved based in the bugs reported by the users. It has a mailing list to ask questions and a mechanism on GitHub to raise any bug issues. As of today, there are close to 342 open issues[8] raised on GitHub and a few of them are high impact bugs. One of the pending bugs is the errors obtained on trying to add vectors to pre-trained model based on FastText.

[7] https://radimrehurek.com/gensim/models/ldamulticore.html
[8] https://github.com/RaRe-Technologies/gensim/issues/

5. Conclusions

Gensim has some limitations in terms of limited support for distributed computing - only some algorithms have support for distributed computing. There are alternative Python packages that provide some of the features that Gensim provides, and in some cases have better performance than Gensim. However, overall, Gensim has been a comprehensive tool for topic modeling and text retrieval and has been widely used for Computer Science research. The package in Python is easy to use and can be used in conjunction with other Python packages for advanced computing.

References

[1] Rehurek, Radim, and Petr Sojka. "Gensim–python framework for vector space modelling." NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic 3.2 (2011).

[2] Haider, Mofiz Mojib, et al. "Automatic Text Summarization Using Gensim Word2Vec and K-Means Clustering Algorithm." 2020 IEEE Region 10 Symposium (TENSYMP). IEEE, 2020.

[3] Shen, Tiancheng, Jia Jia, Yan Li, Yihui Ma, Yaohua Bu, Hanjie Wang, Bo Chen, Tat-Seng Chua, and Wendy Hall. "Peia: Personality and emotion integrated attentive model for music recommendation on social media platforms." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 01, pp. 206-213. 2020.

[4] Nakov, Preslav, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. "SemEval-2017 task 3: Community question answering." arXiv preprint arXiv:1912.00730 (2019).

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

[6] Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." In Advances in neural information processing systems, pp. 3111-3119. 2013.

[7] Joulin, Armand, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. "Bag of tricks for efficient text classification." arXiv preprint arXiv:1607.01759 (2016).

[8] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." the Journal of machine Learning research 3 (2003): 993-1022.

[9] Willett, Peter. "The Porter stemming algorithm: then and now." Program (2006).

[10] Mihalcea, Rada, and Paul Tarau. "Textrank: Bringing order into text." In Proceedings of the 2004 conference on empirical methods in natural language processing, pp. 404-411. 2004.

[11] Text Summarization with Gensim. https://radimrehurek.com/gensim_3.8.3/auto_examples/tutorials/run_summarization.html.

[12] Alami, Nabil, Mohammed Meknassi, and Noureddine En-nahnahi. "Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning." Expert systems with applications 123 (2019): 195-211.

[13] Hofmann, Thomas. "Probabilistic latent semantic indexing." In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 50-57. 1999.

[14] Spacy. https://spacy.io/

[15] Distributed Computing on Gensim. https://radimrehurek.com/gensim/distributed.html

[16] Scikit-learn. https://scikit-learn.org/

[17] Multicore LDA in Python. https://rare-technologies.com/multicore-lda-in-python-from-over-night-to-over-lunch/

[18] Ilyas2020, Sardar Haider Waseem, Zainab Tariq Soomro, Ahmed Anwar, Hamza Shahzad, and Ussama Yaqub. "Analyzing Brexit's impact using sentiment analysis and topic modeling on Twitter discussion." In The 21st Annual International Conference on Digital Government Research, pp. 1-6. 2020.

[19] Maslova, Natalia, and Vsevolod Potapov. "Neural network doc2vec in automated sentiment analysis for short informal texts." In International Conference on Speech and Computer, pp. 546-554. Springer, Cham, 2017.

[20] . SciPy. https://scipy.github.io/devdocs/index.html

[21] NumPy. https://numpy.org/

[22] Doc2Vec. https://radimrehurek.com/gensim/models/doc2vec.html.

[23] Liu, Haixia. "Sentiment analysis of citations using word2vec." arXiv preprint arXiv:1704.00177 (2017).

[24] Le, Quoc, and Tomas Mikolov. "Distributed representations of sentences and documents." In International conference on machine learning, pp. 1188-1196. PMLR, 2014.