Virginia Atieh
Technical Review

CS 410 - Text Information Systems
University of Illinois – UC

Gensim in Python – Features & Applications to the MIR Field

1. Introduction

The human language is the only known form of communication that uses composition, the use of nouns, verbs, and adjectives to express different meanings. If we were to construct a sentence consisting of a subject, verb, and object, with twenty-five different words for each, there would be over fifteen thousand different sentence possibilities. (2) For computers, this level of complexity creates a challenge. This field is referred to as natural language processing (NLP). NLP can be divided into several subtasks: part-of-speech tagging, syntactic parsing, semantic parsing, word sense disambiguation, summarization, sentiment analysis, and more.

2. Music Information Retrieval (MIR)

The field of music information retrieval deals with extracting information from music related sources. The many tasks covered under this term can be categorized under two topics, audio and text. This review will focus on natural language processing of text in MIR. NLP can be used to process music related text such as lyrics, artist biographies, social media, blogs, forums, and encyclopedias. Databases such as MusicBrainz, Discogs, Grove Music as well as the many websites hosting music related content are available for text music information retriveal. Although there are many tools available for NLP tasks that can be applied to the MIR field, we will focus on the Gensim toolkit.

3. Gensim toolkit – Features

There are four core concepts that make up the Gensim toolkit: Document, Corpus, Vector, and Model. The corpus is a collection of documents, which are made up of text, or strings of characters. The models available in Gensim are well known vector space algorithms that have been proven to work in industry applications. These models can be trained by a corpus through data streaming. This provides a space efficient solution to large corpora for training and semantic analysis. Before this can occur, the text documents must be preprocessed. Gensim includes the built in tool *simple_preprocess(doc, deaccent, min_length, max_length)* for such purposes. Vectors can then be generated to represent each document mathematically so that the model can produce meaningful results. The key feature in these models is that they run unsupervised and as such do not require time consuming labeling of documents. The following table outlines built in models included in the Gensim toolkit and their brief explanations. (1)

Model Used in Gensim	**Mechanism**
Term Frequency/Inverse Document Frequency (Tf-Idf)	Utilizes the document’s term frequency in a bag-of-words vector representation to penalize or reward values based on very rare or very common frequencies (IDF). `models.TfidfModel(bow_corpus)`
Latent Semantic Indexing (LSI/LSA)	Begins with either a bag-of-words or TF-IDF representation and reduces the dimensionality by grouping words that occur together. This uncovers meanings to words in a phrase which tend to have less ambiguity than if they were to be analyzed individually. `models.LsiModel(tfidf_corpus, id2word=dictionary, num_topics=300)`
Random Projections (RP)	Introduces randomness to the approximation of similarity between vectors representing documents. Due to its simplicity, this model is memory and CPU efficient in reducing dimensionality while preserving the approximate distance between vectors. `models.RpModel(tfidf_corpus, num_topics=500)`
Latent Dirichlet Allocation (LDA)	Used to discover topics covered in documents. Similar to LSA, LDA reduces the dimension of bag-of-words vectors and employs statistical methods to derive word and topic probabilities. `models.LdaModel(corpus, id2word=dictionary, num_topics=100)`
Hierarchical Dirichlet Process (HDP)	Clusters groups of words (lower dimension) without the need for parameters. `models.HdpModel(corpus, id2word=dictionary)`
Custom Model Provided by user	Gensim AI allows the use of custom vector space model solutions, such as different weighting schemes.

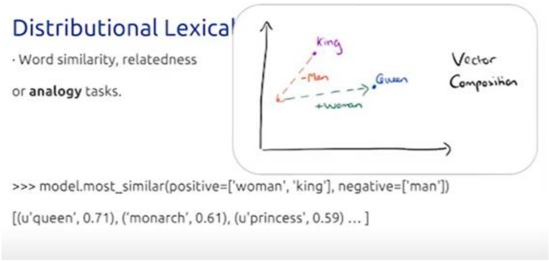
(1)

4. Applications of Gensim in MIR

[Text]

5. Conclusion

[Text]



Distributional Lexical

· Word similarity, relatedness

or **analogy** tasks.

```
>>> model.most_similar(positive=['woman', 'king'], negative=['man'])
[(u'queen', 0.71), ('monarch', 0.61), (u'princess', 0.59) ... ]
```

References

1. “Gensim: Topic Modelling for Humans.” *What Is Gensim? - Gensim*, 30 Aug. 2021, https://radimrehurek.com/gensim/intro.html

2. Pagel, M. Q&A: What is human language, when did it evolve and why should we care?. BMC Biol 15, 64 (2017). https://doi.org/10.1186/s12915-017-0405-3

3. Phon-Amnuaisuk S. (2019) Exploring Music21 and Gensim for Music Data Analysis and Visualization. In: Tan Y., Shi Y. (eds) Data Mining and Big Data. DMBD 2019. Communications in Computer and Information Science, vol 1071. Springer, Singapore. https://doi.org/10.1007/978-981-32-9563-6_1

4. A. M. TURING, I.—COMPUTING MACHINERY AND INTELLIGENCE, Mind, Volume LIX, Issue 236, October 1950, Pages 433–460, https://doi.org/10.1093/mind/LIX.236.433

5. “What Is Music Information Retrieval?¶.” why_mir, https://musicinformationretrieval.com/why_mir.html

6. Choi, Keunwoo & Fazekas, György & Cho, Kyunghyun & Sandler, Mark. (2017). A Tutorial on Deep Learning for Music Information Retrieval.