Michael McClanahan
CS 410 – Text Mining
University of Illinois at Urbana Champaign
(Online) MCS-DS
NetID: mjm31

Technology Review:
Latent Semantic Analysis with Singular Value Decomposition using Gensim

Introduction

Originally developed in 2010 by Radim Rehurek and Petr Sojka, Gensim is a popular open-source Python library for Vector Space Modeling (VSM) of large collections of human text (or documents)[1]. At the time of its creation, its developers sought to improve upon existing Python toolkits by addressing the following issues: (1) a lack of support for unsupervised topic modeling, including Latent Semantic Analysis (LSA), or Latent Semantic Indexing (LSI) and Latent Dirichlet Association, (2) an inability for models produced by existing toolkits to scale properly (due to entire corpora needing to be present in memory), (3) existing libraries not being optimized for natural language processing (NLP), and (4) a lack of one unified toolkit to help users perform modeling with different kinds of NLP and information retrieval (IR) algorithms[1]. Gensim's ability to address these issues, specifically 1 through 3, are exemplified by its LsiModel class, which implements LSA with an optimized, truncated Singular Value Decomposition (SVD) from Brand et al (2006) (1,3). Accordingly, this technology review aims to break down the features of the LsiModel class that accomplish the first two goals of the Gensim project.

Support for unsupervised topic modeling using LSA in the VSM

The inputs for LSA using the VSM are all documents in a corpus, but doing so first requires processing of all documents in a text corpus into bag-of-words representation and then representing the corpus as a document-term matrix, where each column represents the count of a term (or it's TF-IDF score) from the corpus' vocabulary in the document (row) [2]. (As a unified toolkit for NLP/IR tasks, the preparation of the document-term matrix by Gensim objects can be accomplished with only a couple of lines of code. Although convenient, describing this is not the focus of this review). Accordingly, the only required parameter for returning an instance of a LsiModel object is the collection of documents (parameter name: *corpus*) represented as the above document-term matrix.

During the model's initialization, dimensionality reduction is performed against the document-term matrix using an enhanced algorithm for Singular Value Decomposition (SVD). SVD is a linear algebraic technique where a two-dimensional matrix is factorized into the product of three matrices, two of which are orthonormal across the *t* most significant dimensions in the transformed vector space, and one of which is diagonal, consisting of singular values of the original matrix. In LSA for corpus topic modeling, the produced matrices can be thought of as a document-topic matrix, the diagonal matrix, and a term-topic matrix. These matrices can then be used to plot corpus documents and terms in the VSM for comparison, usually by means of cosine similarity scoring. One can compare documents to other documents, terms to other terms, terms to documents (for use in text retrieval following a query), or documents to terms (to summarize the topics of a given document or set of documents). Additionally, because of the

reduced dimensionality, cosine similarities can be preserved at lower computational cost versus plotting the original document-term matrix in the VSM.

Fast, truncated SVD in LsiModel's implementation

Prior to Gensim's implementation, traditional algorithms for incremental SVD, like Gorrell's Generalized Hebbian Algorithm (Gorrell, 2006) [5] had issues in reaching convergence during Expectation-Maximization due computational slowness and internal parameters that were difficult to tune[1]. Instead, LsiModel was implemented purely in Python (*numpy* specifically) using Brand's algorithm for fast incremental SVD updates[1,3]. This algorithm has no internal parameters for users to tune and is much faster due the fact that it represents a pseudo-inverse of a submatrix of an orthogonal matrix, which are the resulting term-topic and document-topic matrix representations of LsiModel's corpus VSM representation[3]. Additionally, Brand's SVD supports the ability to compute a thin SVD of streaming data in a single pass with $O(n)$ time complexity because it implemented an indexing system to keep track of additions and modifications of a resulting SVD matrix.[3] As a result, when LsiModel was first released, it was the only publicly available implementation of LSA that did not require an entire corpus to be loaded into shared memory prior to SVD.[1] This advancement has two very major benefits: (1) Researchers do not need to increase available memory when performing LSA of larger and larger corpora. (2) Incremental updates to the SVD factorized document-term matrix, or LSA model, of a corpus can be distributed across multiple threads and computed in parallel.

Distributed LSA using Gensim[6]

Distributed computation of Gensim's LSA model can be achieved by setting up their Document Similarity Service, which leverages the *Pyro* framework to set up a distributed system for its topic modeling computations. Setting up the system can be done in a few simple steps. At a high level this requires the user to (1) install Gensim with *pickle* Pyro serializers on all nodes in the cluster, (2) run Pryo's name server on one of the machines, (3) run worker scripts on all but one of the machines in the cluster, (4) and choose the remaining node as the LSA dispatcher node by running Gensim's supplied LSA dispatcher script. Once this is complete a LsiModel object can be instantiated on any of the nodes in cluster with the *distributed* parameter set to *True*. For optimal performance, the *chunksize* parameter is provided. It represents the number of documents to be used in each training chunk and, therefore, should be set to a value that evenly distributes documents in the corpus across all workers in the system. For simplicity, all distributed compute (split, compute, and merge) operations are abstracted away by the Document Similarity Service. The user simply sees a LsiModel object returned. Additional distributed NLP and IR tasks can then be performed on the corpus using its representation in the distributed LSA model by initiating them from the previous parent Python process on one of the nodes in the cluster.

Michael McClanahan
CS 410 – Text Mining
University of Illinois at Urbana Champaign
(Online) MCS-DS
NetID: mjm31

Conclusion

In its initial release, Gensim provided a novel implementation of latent semantic analysis in the vector space model for large corpora. Its implementation has proved intuitive for developers and computationally efficient. Its use of a fast, truncated SVD has also allowed for LSA of corpora of any size, regardless of memory limitations, while also enabling easy parallel computation of the LSA model.

References

1. Rehurek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. *Proceedings of LREC 2010 workshop New Challenges for NLP Framework*, 46-50. Retrieved from https://is.muni.cz/publication/884893/lrec2010-rehurek-sojka.pdf

2. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American society for Information science, 41*(6), 391-407.

3. Brand, M. (2006). Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications, 415*(1), 20-30. Retrieved from https://doi.org/10.1016/j.laa.2005.07.021

4. Rehurek, R. (2020, November 4). *models.lsimodel – Latent Semantic Indexing – gensim*. Retrieved from Gensim: Topic modelling for humans: https://radimrehurek.com/gensim/models/lsimodel.html

5. Gorrell, G. (2006). Generalized Hebbian algorithm for incremental Singular Value Decomposition in Natural Languance Processing. *In Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 97-104.

6. Rehurek, R. (2020, November 4). *Document Similarity Server - Gensim*. Retrieved from Gensim: Topic modelling for humans: https://radimrehurek.com/gensim/simserver.html