Independent Study Proposal

Thomas W. Jones

4/29/2018

Background

I am hoping to develop a statistical theory for probabilistic topic models in my dissertation. As such, it is important that I learn the mathematical background and computational implementations of prominent topic models. Two such models are Latent Dirichlet Allocation (LDA), a Bayesian probabilistic topic model, and Probabilistic Latent Semantic Analysis (pLSA), a non-Bayesian probabilistic topic model.

Objectives

Broadly, I see several learning objectives, described below. I also list deliverables which should demonstrate the understandings gained.

1. Learn the foundational computational tools. Deliverable: functions to create core data structures for LDA and pLSA accessible through R.
2. Learn the mathematical foundations of LDA. Deliverable: Summaries of several core papers deriving the LDA model.
3. Learn the computational foundations of LDA. Deliverable: A parallel Gibbs sampler for training LDA models from R.
4. Learn the mathematical foundations of pLSA. Deliverable: A summary of the core paper deriving the pLSA model.
5. Learn the computational foundations of pLSA. Deliverable: A function for training a pLSA model using the EM algorithm from R.

Learn the foundational computational tools

Learning algorithms for LDA and pLSA are computationally-intensive tasks. As such, they benefit greatly from parallelization, where possible. At a minimum, their learning algorithms must be programmed in a low-level language such as C++. Conversely, the R language is much more intuitive for data analyses. However, there is a bridge. In the R language, the packages Rcpp and RcppParallel provide bindings between C++ and R. These tools allow low-level code written in C++ to be called seamlessly from R, a much easier computing enviroment to use. Rcpp works for standard C++, while offering classes and syntatic "sugar" easing R integration. RcppParallel introduces a framework for parallel computing at the C++ level that integrates with R similarlly to Rcpp. I already use Rcpp semi-regularly. Yet I have no experience with RcppParallel.

I propose implementing two simple algorithms as an exercise to learning RcppParallel. Text mining models (such as LDA and pLSA) tend to rely on common data structures, a document term matrix (DTM) or term co-occurrence matrix (TCM). A DTM is a sparse matrix counting the number of times that term j appears in document i. A TCM is a sparse matrix counting the number of times term j and term i occur together within some pre-defined window (e.g. within a whole document, within a 5 term window, etc.) To learn RcppParallel, I will read the e-book published on the RcppParallel website (below) and build parallel implementations to create DTMs and TCMs.

Learn the foundations of LDA

LDA is a Bayesian statistical model. I will read and report on the foundational papers for LDA to understand its mathematical formulation. (As of this writing I am completing a graduate Bayesian statistics course to have the mathematical background to take on this task.)

I will then learn the computational foundations of LDA in two steps. The first is to understand the conventional Gibbs sampling method of training the model. The second is to understand the newer parallel algorithms for approximate Gibbs sampling. Several papers have been published on parallel algorithms. There are also a handful of implementations of parallel Gibbs sampling for LDA. However, none are available in the R language. I hope to fix this by writing a parallel Gibbs sampler accessible from R using RcppParallel.

Learn the foundations of pLSA

pLSA is a non-Bayesian (frequentist) statistical model. It never gained the popularity of LDA, but it is an important model in the topic modeling taxonomy. As with LDA, I will read the foundational paper and report my understanding of the mathematical model. I then hope to use Rcpp or RcppParallel to implement training for pLSA accesible from R.

An added benefit

I am the author of an R package for text mining, textmineR. However, because of limitations in my own knowledge, I have been reliant on others' code in textmineR's code base. This creates two issues: 1) More dependencies mean more complexity and higher chance of some component failing based on a user's system or API-altering updates in a dependency. 2) Some dependencies could (potentially) be sped up, increasing usability of the package.

I intend to integrate the software deliverables from this project into textmineR's code base. This not only helps me, but it also helps the R user community by giving them access to (hopfully) fast and accurate implementations of popular topic models. At present, I do not posess the knowlede or ability to write these myself. I am hoping this independent study will teach me.

I am also planning to use my summaries of the research papers as a portion of a literature review in my dissertation, once I have acheived candidacy.

Organization

I imagine this project being conducted in three phases. I intend to spend the majority of my time on LDA as it is the most technically challenging model. If time remains in the study period, I have a bonus phase consisting of reading about and summarizing more topic models.

Phase 1 (est. 3 weeks)

1. Background reading on RcppParallel
2. Design algorithms for DTM and TCM creation
3. Implement algorithms and integrate into textmineR

Phase 2 (est. 8 weeks)

1. Background reading on LDA (mathematical theory)

2. Background reading on LDA implementations (focus on Gibbs sampling and parallel Gibbs)
3. Design my own parallel Gibbs sampling algorithm
4. Implement this algorithm in RcppParallel and integrate into textmineR

Phase 3 (est. 4 weeks)

1. Background reasing on pLSA
2. Design a pLSA algorithm
3. Implement pLSA algorithm and integrate into textmineR
4. Explore possibility of parallelization

Bonus phase (est. n weeks)

1. Gather papers for the following topic models:

- Supervised LDA
- Correlated topic models
- Structured topic models
- Global vectors for word representations
- word2vec

2. Read and summarize those papers

Deliverables

1. All implemented algorithms will be released as open source software as part of the textmineR package
2. For each paper I read, I will write a summary of its core concepts

Tentative reading list

RcppParallel

1. RcppParallel's web book, including examples

Latent Dirichlet Allocation

1. Latent Dirichlet Allocation, Blei et al., JMLR (3), 2003.
2. Womack, A., Moreno, E., & Casella, G. (2013). Consistency in Latent Allocation Models. Submitted to Annals of Statistics.
3. Finding scientific topics, Griffiths and Steyvers, PNAS (101), 2004.
4. Fast collapsed gibbs sampling for latent dirichlet allocation, Porteous et al., KDD 2008.
5. Distributed Inference for Latent Dirichlet Allocation, Newman et al., NIPS 2007.
6. PLDA: Parallel Latent Dirichlet Allocation for Large-scale Applications. Yi Wang, Hongjie Bai, Matt Stanton, Wen-Yen Chen, and Edward Y. Chang. AAIM 2009.
7. PLDA+: Parallel Latent Dirichlet Allocation with Data Placement and Pipeline Processing. Zhiyuan Liu, Yuzhou Zhang, Edward Y. Chang, Maosong Sun. ACM Transactions on Intelligent Systems and Technology, special issue on Large Scale Machine Learning. 2011.

Probabilistic Latent Semantic Analysis

1. Hofmann, T. (1999, July). Probabilistic latent semantic analysis. In Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence (pp. 289-296). Morgan Kaufmann Publishers Inc.

Additional Topic models

1. Mcauliffe, J. D., & Blei, D. M. (2008). Supervised topic models. In Advances in neural information processing systems (pp. 121-128).
2. Blei, D. M., & Lafferty, J. D. (2005, December). Correlated topic models. In Proceedings of the 18th International Conference on Neural Information Processing Systems (pp. 147-154). MIT Press.
3. Wallach, H. M. (2008). Structured topic models for language (Doctoral dissertation, University of Cambridge).
4. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).
5. Goldberg, Y., & Levy, O. (2014). word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method. arXiv preprint arXiv:1402.3722.