Distributed Latent Dirichlet Allocation via Tensor Factorization

Furong Huang
University of California Irvine
furong@uci.edu

Sergiy Matusevych
Microsoft CISL
sergiym@microsoft.com

Anima Anandkumar
University of California Irvine
a.anandkumar@uci.edu

Nikos Karampatziakis, Paul Mineiro
Microsoft CISL
{nikosk,pmineiro}@microsoft.com

Abstract

We describe a distributed implementation for Latent Dirichlet Allocation parameter estimation based upon the method of moments.

1 Introduction

Latent Dirichlet Allocation (LDA) has proven extremely popular and versatile since its introduction over a decade ago. LDA is successful in part because it assigns a mixture of latent states ("topics") to each set of exchangeable observations ("document"), in contrast to a hard clustering. This property complicates the estimation of latent parameters, and has led to extensive research in disparate learning techniques. Broadly speaking there are 3 basic strategies: variational inference [3]; Markov chain Monte Carlo [6]; and the method of moments [1], the latter having been recently discovered.

Due to high dimensional data with large vocabulary size; numerous documents; and number of topics, computational constraints are the limiting factor to developing large scale topic models. This has motivated research into scalable computational strategies for LDA. In the single node context, stochastic variational inference [9] is fast and accurate, but has high communication costs in the distributed setting. Batch variational inference has a more favorable ratio of communication to computation as the E-step (but not the M-step) is embarrisingly parallel [14]. Markov chain Monte Carlo (MCMC) techniques have also been implemented in the distributed setting, both synchronous [16, 21] and asynchronous [17] variants.

Due to their recent introduction, there are no distributed implementations of method of moments based approaches to LDA. We leverage that the method of moments for LDA reduces to canonical polyadic (CP) decomposition of a tensor, a problem which has received extensive study in the literature [11], including distributed variants [10]. We combine ALS with whitening preprocessing (data orthogonalization and dimensionality reduction) motivated by better convergence rate and perturbation guarantees [1] compared to previous methods. Additionally, the preprocessing has the benefit that the subsequent tensor decomposition is independent of the vocabulary size and the number of documents.

Although ALS requires many iterations to converge (more than would be tolerable using map-reduce without custom support for low-overhead iteration), we utilize REEF [4], a distributed processing framework which runs on YARN [19] managed clusters, e.g., a Hadoop 2 installation.

2 LDA Moment Characterization

Because our algorithm is based upon spectral methods for LDA, we review the relevant background material here. LDA models each of n documents as a mixture over k latent topics, where each topic

Algorithm 1 Distributed Spectral LDA Parameter Estimation

function LDA($k, \alpha_0, D \in \mathbb{R}^{n \times d}$)
  `// Whiten` $\hat{M}_2$ `(3 data passes)`
  $(U, \Sigma, V) = \mathrm{svd}(\hat{M}_2(D), k)$. ▷ $U \in \mathbb{R}^{n \times k}$
  `// Compute projected` $\hat{M}_1$ `(1 map-reduce pass)`
  $\hat{M}_1 \leftarrow \mathrm{mean}(U, \Sigma)$ ▷ $\hat{M}_1 \in \mathbb{R}^k$
  `// Compute projected` $\hat{M}_3$ `matricization (1 map-reduce pass)`
  $\hat{M}_3 \leftarrow \mathrm{compute}M_3(U, \Sigma, \hat{M}_1)$ ▷ $\hat{M}_3 \in \mathbb{R}^{k \times k^2}$
  `// CP decompose` $\hat{M}_3$ `via ALS (multiple BSP iterations)`
  $(\lambda, A) \leftarrow \mathrm{cp\_als}(\hat{M}_3, k)$ ▷ $\lambda \in \mathbb{R}^k, A \in \mathbb{R}^{k \times k}$
  `// Recover LDA parameters from factorization (single node)`
  $\{\beta_i\}_{i=1}^k \leftarrow \mathrm{unproject}(A)$
  $\alpha_i \propto \lambda_i^{-2}$ ▷ see Section 4.3 of [1]
  return $(\{\beta_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k)$
end function

defines a multinomial topic-word conditional emission probability β over d tokens. The mixing distribution of latent topics per document is modeled as drawn from a Dirichlet hyperprior. The parameters of interest to estimate from a corpus are the topic-token emission probabilities for each topic $\{\beta_i\}_{i=1}^k \in \Delta^d$, and the Dirichlet hyperprior parameter $\alpha \in \Delta^k$. A key result is Theorem 3.5 of [1], which states that the shifted moments of a rank-k LDA model given by

$$M_1 \stackrel{\text{def}}{=} \mathbb{E}[x_1], \tag{1}$$

$$M_2 \stackrel{\text{def}}{=} \mathbb{E}[x_1 \otimes x_2] - \frac{\alpha_0}{\alpha_0 + 1} M_1 \otimes M_1, \tag{2}$$

$$\begin{aligned} M_3 \stackrel{\text{def}}{=} & \ \mathbb{E}[x_1 \otimes x_2 \otimes x_3] \\ & - \frac{\alpha_0}{\alpha_0 + 2} \left(\mathbb{E}[x_1 \otimes x_2 \otimes M_1] + \mathbb{E}[x_1 \otimes M_1 \otimes x_3] + \mathbb{E}[M_1 \otimes x_2 \otimes x_3] \right) \\ & + \frac{\alpha_0^2}{(\alpha_0 + 2)(\alpha_0 + 1)} M_1 \otimes M_1 \otimes M_1, \end{aligned} \tag{3}$$

are related to the latent parameters via

$$M_2 = \sum_{i=1}^k \alpha_i \, \beta_i \otimes \beta_i, \quad M_3 = \sum_{i=1}^k \alpha_i \, \beta_i \otimes \beta_i \otimes \beta_i.$$

Here x_1, x_2, and x_3 are tokens in the same document. Hyperprior vector α is identifiable to its direction but not amplitude, therefore we specify hyperparameter $\alpha_0 \stackrel{\text{def}}{=} \sum_{i=1}^k \alpha_i$. Heuristically, a small α_0 will prefer documents that have only a few topics.

This result indicates that a tensor decomposition of M_3 reveals the latent parameters of interest. Although M_3 need not be explicitly formed to perform the decomposition, for latent dimensionalities typically employed with LDA (e.g., $k < 10^4$), it is advantageous to explicitly form the empirical $M_3 \in \mathbb{R}^{k \times k \times k}$ in a reduced dimensional space. This makes subsequent computation independent of both the number of documents and the number of tokens. Note that the reduced dimensional representation is theoretically correct due to the moment characterization indicating that a low-rank decomposition of M_2 is sufficient to identify the subspace containing the $\{\beta_i\}$. Furthermore, it is practically efficient because randomized SVD[7] provides an inexpensive way to obtain a low-rank decomposition of M_2.

3 Algorithm

Algorithm 1 is our distributed spectral LDA algorithm for parameter estimation. The major phases are whiten, project, tensor decompose, and unproject. Not shown is our algorithm for estimating the latent state distribution for a document given the estimated model parameters: for this we utilize the variational lower bound strategy from [3]. This is only used at test time to evaluate perplexity on held-out documents, and from a systems perspective is a map-only embarrassingly parallel function given the model parameters, so we omit detailed discussion.

The rank-k SVD on line 3 of algorithm 1 is done in three map-reduce [5] data passes via randomized techniques [7]. Significant efficiency gains are possible via analytical composition of the fast empirical count estimation formulas in section 6.1 of [1] with the randomized SVD algorithm. For example, the action of $\mathbb{E}[x_1 \otimes x_2]$ on a basis Ω simplifies to

$$\mathbb{E}[x_1 \otimes x_2]\Omega = \frac{1}{n} \sum_{m=1}^{n} \left(\frac{1}{\binom{l_m}{2}} \frac{1}{2!} \sum_{i | c_{m,i} \neq 0} \left(c_m^\top \Omega - \Omega_i \right) c_{m,i} \vec{e}_i \right),$$

where c_m is the vector of token counts for document m, $l_m = \sum_i c_{m,i}$, and $\vec{e}_i$ is the i^{th} basis vector.

Once the high-dimensional data has been projected into the compact k dimensional space, the empirical mean $\hat{M}_1 \in \mathbb{R}^k$ and empirical (matricized) shifted third moment $\hat{M}_3 \in \mathbb{R}^{k \times k^2}$ are computed in the projected space using equations (1) and (3). This can be done with 2 additional map-reduce data passes.

After forming $\hat{M}_3$, tensor decomposition via ALS proceeds. This involves solving a sequence of least squares problems of the form

$$(\lambda, A) \leftarrow \min_{\sigma \in \mathbb{R}^k, X \in \mathbb{R}^{k \times k}} \left\| X \operatorname{Diag}(\sigma) \left(C \odot B \right)^\top - \hat{M}_3 \right\|^2 \text{s.t. } \forall k : \|X_k\| = 1, \sigma_k \geq 0 \qquad (4)$$

where $\odot$ denotes Khatri-Rao product. ALS alternatively optimizes for A, B, and C, each time estimating λ to ensure eigenvectors are normalized [11]. For such problems the formula in Theorem 2 of [12] is highly useful: $(C \odot B)^\dagger = \left(\left(C^\top C \right) \star \left(B^\top B \right) \right)^\dagger (C \odot B)^\top$, where $\star$ denotes element-wise product. Additional efficiency is via possible via “Fast Property 2” of [13], which states $\left((C \odot B)^\top x \right)_i = B_i^\top X C_i$, where $X = \operatorname{reshape}(x, k, k)$.

After the tensor decomposition has converged, we map the eigenvectors back into the original token space. We employ the following novel strategy based upon the inverse whitening transformation in Theorem 4.3 of [1] combined with enforcing a simplex constraint. Specifically, given the matrix $A \in \mathbb{R}^{k \times k}$ of reduced dimensionality eigenvectors we wish to find a matrix $\Phi \in \mathbb{R}^{d \times k}$ such that $\Phi \approx (W^\top)^\dagger A \operatorname{Diag}(\lambda)$ where $W = \Sigma^\dagger V \in \mathbb{R}^{k \times d}$ is the whitening matrix from the SVD from line 3 of algorithm 1. Furthermore we require each column of Φ to be on the simplex. This can be done via ADMM as a post-processing step on a single node, by iterating the following equations

$$\begin{aligned}
\Phi &\leftarrow \arg\min_{\Theta} \left\| \Theta - (W^\top)^\dagger A \operatorname{Diag}(\lambda) \right\|_2^2 + \frac{\rho}{2} \|\Theta + U - Z\|_2^2 \\
&\leftarrow \frac{1}{\rho + 1} \left((W^\top)^\dagger A \operatorname{Diag}(\lambda) + \rho (Z - U) \right) \\
Z &\leftarrow \Pi_\Delta (\Phi + U), \qquad (5) \\
U &\leftarrow \Phi + U - Z.
\end{aligned}$$

Equation (5) is a minimum Euclidean norm projection of each column onto the simplex and can be done in $O(kd \log d)$ time [20].

3.1 ALS implementation details

From a systems standpoint, the ALS computation conforms to a Bulk Synchronous Parallel computation model [18]. Each column of the left hand side of equation (4) can be estimated independently on a portion of $\hat{M}_3$, providing up to degree k parallelism. Processors must then synchronize by exchanging newly estimated columns before proceeding to the next least squares subproblem. The space requirements for each worker is $O(k^2)$, and the amount communicated to each worker per ALS iteration is $O(k^2)$.

We map the BSP structure onto REEF via a master-worker arrangement of evaluators, where workers perform concurrent computations, and the singleton master organizes communication and provides

Nodes	Training Time (s)
1	3475
5	1860
10	1392
20	1184
40	1289

Table 1: Running times for $k = 100$ as number of nodes is varied.

barrier synchronization. Each worker is responsible for a row slice of the estimated factor A using a row slice of $\hat{M}_3$ and the full other factors B and C. Workers iteratively recompute their updated factor, broadcast to other workers, and receive updated factors from other workers, where the last step provides synchronization. The master checks for the termination condition and optionally halts during the synchronization phase. This arrangement facilitates fault-mitigation strategies, e.g., the master can detect a fault in a worker, instruct the remaining workers to continue with a partial factor update while requesting more evaluators and reconstructing the communication mesh.

4 Experimental Results

We describe results with the New York Times news corpus [15], obtained via the UCI Machine Learning repository [2]. This corpus has 300,000 documents, 102,660 unique tokens, and roughly 100M total tokens. We evaluate model quality using average per-token log perplexity [3],

$$\log \mathrm{perplexity}(\{c_m\}_{m=1}^{n}) = -\frac{\sum_{m=1}^{n} \log p(c_m)}{\sum_{m=1}^{n} \sum_{w=1}^{d} c_{m,i}}.$$

To estimate $\log p(c_m)$ for a single document we use the variational lower bound from [8].

Using the stochastic variational inference implementation from [8] with $k = 100$, we obtain a training log perplexity of 8.31 and a training running time of 2 hours. Using spectral inference we obtain training log perplexity of 8.13 with $k = 100$. Running times depend upon the number of compute nodes utilized, as indicated in table 1. Initially more nodes are beneficial, but eventually communication overhead dwarfs increased processing power and training times do not decrease with additional nodes.

5 Conclusion

We have presented a distributed implementation of LDA based upon the method of moments. The core of the algorithm is the factorization of a tensor which we perform with Alternating Least Squares. To this end we leveraged REEF, a distributed processing library on top of YARN, which enables the implementation of iterative algorithms with low overhead between iterations. In the future, we plan to apply the same ideas in other applications of mixed membership models such as community detection where we expect to have an even bigger advantage compared to variational methods.

References

[1] Anima Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*, 2012.

[2] K. Bache and M. Lichman. UCI machine learning repository, 2013.

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

[4] Byung-Gon Chun, Tyson Condie, Carlo Curino, Chris Douglas, Sergiy Matusevych, Brandon Myers, Shravan Narayanamurthy, Raghu Ramakrishnan, Sriram Rao, Josh Rosen, et al. Reef: Retainable evaluator execution framework. *Proceedings of the VLDB Endowment*, 6(12):1370–1373, 2013.

[5] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[6] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences of the United States of America*, 101(Suppl 1):5228–5235, 2004.

[7] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructi ng approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[8] Matthew Hoffman, Francis R. Bach, and David M. Blei. Online learning for latent dirichlet allocation. In J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 856–864. Curran Associates, Inc., 2010.

[9] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[10] U Kang, Evangelos Papalexakis, Abhay Harpale, and Christos Faloutsos. Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 316–324. ACM, 2012.

[11] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[12] Shuangzhe Liu and Gõtz Trenkler. Hadamard, khatri-rao, kronecker and other matrix products. *Int. J. Inform. Syst. Sci*, 4(1):160–177, 2008.

[13] Charles F. Van Loan. Lecture 5: The CP representation and tensor rank. University Lecture, 2010.

[14] Ramesh Nallapati, William Cohen, and John Lafferty. Parallelized variational em for latent dirichlet allocation: An experimental evaluation of speed and scalability. In *Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on*, pages 349–354. IEEE, 2007.

[15] David Newman, Chaitanya Chemudugunta, Padhraic Smyth, and Mark Steyvers. Analyzing entities and topics in news articles using statistical topic models. In *Intelligence and Security Informatics*, pages 93–104. Springer, 2006.

[16] David Newman, Padhraic Smyth, Max Welling, and Arthur U Asuncion. Distributed inference for latent dirichlet allocation. In *Advances in neural information processing systems*, pages 1081–1088, 2007.

[17] Alexander Smola and Shravan Narayanamurthy. An architecture for parallel topic models. *Proceedings of the VLDB Endowment*, 3(1-2):703–710, 2010.

[18] Leslie G Valiant. A bridging model for parallel computation. *Communications of the ACM*, 33(8):103–111, 1990.

[19] Vinod Kumar Vavilapalli, Arun C Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves, Jason Lowe, Hitesh Shah, Siddharth Seth, et al. Apache hadoop yarn: Yet another resource negotiator. In *Proceedings of the 4th annual Symposium on Cloud Computing*, page 5. ACM, 2013.

[20] Weiran Wang and Miguel A Carreira-Perpinán. Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application. *arXiv preprint arXiv:1309.1541*, 2013.

[21] Yi Wang, Hongjie Bai, Matt Stanton, Wen-Yen Chen, and Edward Y Chang. Plda: Parallel latent dirichlet allocation for large-scale applications. In *Algorithmic Aspects in Information and Management*, pages 301–314. Springer, 2009.