The Journal of Open Source Software

tidylda: An R Package for Latent Dirichlet Allocation Using 'tidyverse' Conventions

Tommy Jones [1*]

1 Foundation, USA * These authors contributed equally.

DOI: 10.21105/joss.06800

Software

- Review
- Repository
- Archive

Editor: Kanishka B. Narayan

Reviewers:

- @maximelenormand
- @hassaniazi

Submitted: 16 May 2024
Published: 24 July 2024

Summary

tidylda is a package for a topic model, Latent Dirichlet Allocation or LDA (Blei et al., 2003), that is natively compatible with the *tidyverse* (Wickham et al., 2019). *tidylda*'s Gibbs sampler is written in C++ for performance and offers several novel features, such as transfer learning for LDA using the tLDA model. It also has methods for sampling from the posterior of a trained model, for more traditional Bayesian analyses.

Statement of Need

Packages that implement topic models in R and other languages are plentiful. Why do we need another? *tidylda*'s native compatibility with the *tidyverse* makes it significantly more user friendly than other topic modeling packages. It also enables more traditional Bayesian analyses such as the ability to set more flexible priors, burn-in iterations, averaging over segments of the Gibbs sample chain, and sampling from the posterior that other packages lack. Finally, *tidylda* implements a transfer learning algorithm developed in (Jones, 2023), unavailable in any other package and described in more detail in the following section.

The "tidyverse" family of packages for R

tidylda takes its syntactic cues from an ecosystem of R packages known as *the tidyverse*. The tidyverse's goal is to "facilitate a conversation between a human and computer about data" (Wickham et al., 2019). Packages in—and adjacent to—the tidyverse share a common design philosophy and syntax based on "tidy data" principles (Wickham & others, 2014). Tidy data has each variable in a column, each observation in a row, and each observational unit in a table. Extensions include the *broom* package (Robinson, 2014) for "tidying" up outputs from statistical models and the in-development *tidymodels* ecosystem (Khun & Wickham, 2018) which extends the tidyverse philosophy to statistical modeling and machine learning workflows.

Silge and Robinson articulated a "tidy data" framework for text analyses—the *tidytext* package (Silge & Robinson, 2016). Their approach has "one row per document per token". The *tidytext* package provides functionality to tokenize a corpus, transform it into this "tidy" format, and manipulate it in various ways, including preparing data for input into some of R's many topic modeling packages. The *tidytext* package also provides tidying functions in the style of the *broom* package, which harmonizes outputs from some of R's topic modeling packages into more usable formats. *tidylda* manages inputs and outputs in the flavor of *tidytext* but in one self contained package.

Topic modeling software in R

R has many packages for topic modeling; none are natively "tidy" though some have wrapper functions available in *tidytext* that produce tidy outputs. In almost all cases these models support only scalar, or "symmetric", priors for topics over documents.

The *textmineR* package (Jones, 2015) is *tidylda*'s predecessor, supporting vector, or "asymmetric"," priors. It supports fitting several topic models, not just LDA. But *textmineR* does not support transfer learning nor is it consistent with the *tidyverse* principles.

The *topicmodels* package (Grün & Hornik, 2011) supports fitting models for LDA and correlated topic models (Blei & Lafferty, 2007) with both a collapsed Gibbs sampler and variational expectation maximization (VEM). When using VEM, α may be treated as a free parameter and estimated during fitting. It only allows users to set symmetric priors. It is designed to be interoperable with the *tm* package (Feinerer et al., 2008), the oldest framework for text analysis in R. *tidytext* provides "tidier" functions to make the *topicmodels* package interoperable with other frameworks, such as *quanteda* (Benoit et al., 2018), *text2vec* (Selivanov et al., 2020), and more.

The *lda* package (Chang, 2015) provides a collapsed Gibbs sampler for LDA, and other less well-known models. Its Gibbs sampler is one of the fastest. It allows users to set only symmetric priors. Its syntax is esoteric and it requires text documents as input, but does not offer much flexibility in the way of pre-processing. It is generally not interoperable with other packages without significant programming on the part of its users.

The *text2vec* package (Selivanov et al., 2020) is a framework for very fast text pre-processing and modeling. *text2vec* implements LDA using the WarpLDA algorithm (Chen et al., 2015), but it only allows symmetric priors. *text2vec* also offers other models related to distributional semantics. Its syntax is also esoteric using R's *R6* objects that reach back to actively running C++ code for performance reasons. One of *text2vec*'s novel features is that it implements many different coherence calculations; most packages implement only one or none.

The *STM* package (Margaret E. Roberts et al., 2019) implements VEM algorithms for structural topic models (Margaret E. Roberts et al., 2013) and correlated topic models (Blei & Lafferty, 2007). *STM* is well-supported with interfaces in *tidytext*. It offers unique capabilities for model initialization somewhat analogous to transfer learning. Models may be initialized at random or from an LDA model that has run for a few iterations. *STM* does not offer this as a fully-fledged "transfer learning" paradigm. Instead it is a flag the user sets at run time. *STM* then produces the LDA model to hand off to the STM model internally. STM has several unique methods for setting priors but the documentation makes it appear that they are all symmetric.

Latent Dirichlet Allocation and Notation

LDA is a Bayesian latent variable model for text (Blei et al., 2003). It decomposes a data set of word counts, X, whose row/column entries, d, v, represent the number of times word v is found in document d, into two matrices: Θ and B. The former gives a distribution of (latent) topics over documents and the latter gives a distribution of words over topics. Formally, LDA is

$$z_{d_n} | \theta_d \sim \text{Categorical}(\theta_d) \tag{1}$$

$$w_{d_n} | z_k, \beta_k^{(t)} \sim \text{Categorical}(\beta_k^{(t)}) \tag{2}$$

$$\theta_d \sim \text{Dirichlet}(\alpha) \tag{3}$$

$$\beta_k^{(t)} \sim \text{Dirichlet}(\eta) \tag{4}$$

where random variables w_{d_n} and z_{d_n} represent the word and topic of the n-th word of the d-th document. The user sets prior values for α and η as well as specifying the number of topics, K.

Posterior estimates of Θ and B along with the data, X, allow for the calculation of Λ. Where $\beta_{k,v}$ is $P(\text{word}_v|\text{topic}_k)$, $\lambda_{k,v}$ is $P(\text{topic}_k|\text{word}_v)$.

One of the common ways of estimating LDA is through collapsed Gibbs sampling. Gibbs sampling is a Markov chain Monte Carlo method for estimating parameters of a probability distribution where a closed form solution does not exist or is computationally intractable. In the background, the sampler tracks the number of times topics are sampled with two matrices: Cd and Cv. The former's row/column entries, d, k, are the number of times topic k was sampled in document d. The latter's row/column entries, k, v are the number of times topic k was sampled for word v.

Transfer LDA (tLDA)

Formally, tLDA modifies LDA in the following way:

$$\beta_k^{(t)} \sim \text{Dirichlet}(\omega_k^{(t)} \cdot \mathbb{E}\left[\beta_k^{(t-1)}\right]) \tag{5}$$

The above indicates that tLDA places a matrix prior for words over topics where $\eta_{k,v}^{(t)} = \omega_k^{(t)} \cdot \mathbb{E}\left[\beta_{k,v}^{(t-1)}\right] = \omega_k^{(t)} \cdot \frac{Cv_{k,v}^{(t-1)} + \eta_{k,v}^{(t-1)}}{\sum_{v=1}^{V} Cv_{k,v}^{(t-1)}}$. Because the posterior at time t depends only on data at time t and the state of the model at time $t-1$, tLDA models retain the Markov property. Rather than K tuning weights, $\omega_k^{(t)}$, users tune a single parameter, a.

When $a^{(t)} = 1$, fine tuning is equivalent to adding the data in $X^{(t)}$ to $X^{(t-1)}$. In other words, each word occurrence in $X^{(t)}$ carries the same weight in the posterior as each word occurrence in $X^{(t-1)}$. When $a^{(t)} < 1$, then the posterior has recency bias. When $a^{(t)} > 1$, then the posterior has precedent bias. Each word occurrence in $X^{(t)}$ carries less weight than each word occurrence in $X^{(t-1)}$.

For more details on tLDA see (Jones, 2023).

tidylda's Novel Features

Model Initialization and Gibbs Sampling

tidylda's Gibbs sampler has several unique features, described below.

Non-uniform initialization: Most LDA Gibbs samplers initialize by assigning words to topics and topics to documents by sampling from a uniform distribution. This ensures initialization without incorporating any prior information. *tidylda* incorporates the priors in its initialization. It begins by drawing $P(\text{topic}|\text{document})$ and $P(\text{word}|\text{topic})$ from Dirichlet distributions with parameters α and η, respectively. Then *tidylda* uses the above probabilities to construct $P(\text{topic}|\text{word}, \text{document})$ and makes a single run of the Gibbs sampler to initialize two matrices tracking topics over documents and words over topics, denoted Cd and Cv, respectively.

This non-uniform initialization powers tLDA, described above, by starting a Gibbs run near where the previous run left off. For initial models, it uses the user's prior information to tune where sampling starts.

Flexible priors: *tidylda* has multiple options for setting LDA priors. Users may set scalar values for α and η to construct symmetric priors. Users may also choose to construct vector priors for both α and η for a full specification of LDA. Additionally, *tidylda* allows users to set a

matrix prior for η, enabled by its implementation of tLDA. This enables users to set priors over word-topic relationships informed by expert input. The best practices for encoding expert input in this manner are not yet well studied. Nevertheless, this capability makes *tidylda* unique among LDA implementations.

Burn in iterations and posterior averaging: Most LDA Gibbs samplers construct posterior estimates of Θ and B from Cd and Cv's values of the final iteration of sampling, effectively using a single sample. This is inconsistent with best practices from Bayesian statistics, which is to average over many samples from a stable posterior. *tidylda* enables averaging across multiple samples of the posterior with the `burnin` argument. When `burnin` is set to a positive integer, *tidylda* averages the posterior across all iterations larger than `burnin`. For example, if `iterations` is 200 and `burnin` is 150, *tidylda* will return a posterior estimate that is an average of the last 50 sampling iterations. This ensures that posterior estimates are more likely to be representative than any single sample.

Transfer learning with tLDA: Finally, and as discussed previously, *tidylda*'s Gibbs sampler enables transfer learning with tLDA.

Tidy Methods

tidylda's construction follows *Conventions of R Modeling Packages* (Khun, 2019). In particular, it contains methods for `print`, `summary`, `glance`, `tidy`, and `augment`, consistent with other "tidy" packages. These methods are briefly described below.

- `print`, `summary`, and `glance` return various summaries of the contents of a *tidylda* object, into which an LDA model trained with *tidylda* is stored.
- `tidy` returns the contents of Θ, B, or Λ (stored as `theta`, `beta`, and `lambda` respectively), as specified by the user, formatted as a tidy `tibble`, instead of a numeric matrix.
- `augment` appends model outputs to observational-level data. Taking the cue from *tidytext* (Fay, 2018), "observational-level" data is one row per word per document. Therefore, the key statistic used by `augment` is $P(\text{topic}|\text{word}, \text{document})$. *tidylda* calculates this as $\Lambda \times P(\text{word}|\text{document})$, where $P(\text{word}|\text{document}_d) = \frac{x_d}{\sum_{v=1}^{V} x_{d,v}}$.

Posterior Methods

tidylda enables traditional Bayesian uncertainty quantification by sampling from the posterior. The posterior distribution for θ_d is Dirichlet$(Cd_d + \alpha)$ and the posterior distribution for β_k is Dirichlet$(Cv_k + \eta)$ (or Dirichlet$(Cv_k + \eta_k)$ for tLDA). *tidylda* enables a `posterior` method for *tidylda* objects, allowing users to sample from the posterior to quantify uncertainty for estimates of estimated parameters.

tidylda uses one of two calculations for predicting topic distributions (i.e., $\hat{\theta}_d$) for new documents. The first, and default, is to run the Gibbs sampler, constructing a new Cd for the new documents but without updating topic-word distributions in B. The second uses a dot product, $X^{(new)} \cdot \Lambda'$, where the rows of $X^{(new)}$ are normalized to sum to 1. *tidylda* actually uses the dot product prediction combined with the *non-uniform initialization*—described above—to initialize Cd when predicting using the Gibbs sampler.

Other details

You can install the development version of *tidylda* from GitHub here or the CRAN release here. Instructions for both are in the *tidylda* repository's README file.

tidylda's repository and CRAN release contain several vignettes on usage and background. Most of the vignette content is included in this paper. One exception is the coherence calculation used in *tidylda*. The PDF version of that vignette is available on CRAN here.

Acknowledgements

Many people over the years have supported the development of *tidylda*. But most notably are

- Wil Doane, for making me a better programmer and giving ample good advice.
- Brendan Knapp, for helping with the C++ code.
- Barum Park, whose code formed the basis of the multinomial sampler in C++.
- My PhD committee, without whom *tidylda* would be full of "good ideas", but not peer-reviewed research.

References

Benoit, K., Watanabe, K., Wang, H., Nulty, P., Obeng, A., Müller, S., & Matsuo, A. (2018). quanteda: An R package for the quantitative analysis of textual data. *Journal of Open Source Software*, *3*(30), 774. https://doi.org/10.21105/joss.00774

Blei, D. M., & Lafferty, J. D. (2007). A correlated topic model of Science. *The Annals of Applied Statistics*, *1*(1), 17–35. https://doi.org/10.1214/07-aoas114

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, *3*.

Chang, J. (2015). *lda*. https://CRAN.R-project.org/package=lda

Chen, J., Li, K., Zhu, J., & Chen, W. (2015). WarpLDA: a Cache Efficient O(1) Algorithm for Latent Dirichlet Allocation. *arXiv*.

Fay, C. (2018). Text Mining with R: A Tidy Approach. *Journal of Statistical Software*, *83*(Book Review 1). https://doi.org/10.18637/jss.v083.b01

Feinerer, I., Hornik, K., & Meyer, D. (2008). Text Mining Infrastructure in R. *Journal of Statistical Software*, *25*(5).

Grün, B., & Hornik, K. (2011). topicmodels: An R Package for Fitting Topic Models. *Journal of Statistical Software*, *40*(13).

Jones, T. (2015). *textmineR: Functions for Text Mining and Topic Modeling*. https://CRAN.R-project.org/package=textmineR

Jones, T. (2023). *Latent dirichlet allocation for natural language statistics* [PhD thesis]. George Mason University.

Khun, M. (2019). *Conventions for r modeling packages*. https://tidymodels.github.io/model-implementation-principles/index.html

Khun, M., & Wickham, H. (2018). *Tidymodels*. https://www.tidymodels.org/

Roberts, Margaret E., Stewart, B. M., & Tingley, D. (2019). stm : An R Package for Structural Topic Models. *Journal of Statistical Software*, *91*(2). https://doi.org/10.18637/jss.v091.i02

Roberts, Margaret E., Stewart, B. M., Tingley, D., & Airoldi, E. M. (2013). *The Structural Topic Model and Applied Social Science*.

Robinson, D. (2014). broom: An R Package for Converting Statistical Analysis Objects Into Tidy Data Frames. *arXiv*.

Selivanov, D., Bickel, M., & Wang, Q. (2020). *text2vec*. CRAN. https://CRAN.R-project.org/package=text2vec

Silge, J., & Robinson, D. (2016). tidytext: Text Mining and Analysis Using Tidy Data Principles in R. *The Journal of Open Source Software*, *1*(3), 37. https://doi.org/10.21105/joss.00037

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund,

G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., … Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Wickham, H., & others. (2014). Tidy data. *Journal of Statistical Software*, *59*(10), 1–23.