Journal of Machine Learning Research 18 (2017) 1-5 Submitted 8/15; Revised 9/16; Published 3/17

Refinery: An Open Source Topic Modeling Web Platform

Daeil Kim DAEIL@CS.BROWN.EDU
Benjamin F. Swanson DUJIAOZHU@GMAIL.COM
Michael C. Hughes MHUGHES@CS.BROWN.EDU
Erik B. Sudderth SUDDERTH@CS.BROWN.EDU
Department of Computer Science, Brown University, Providence, RI 02192, USA

Editor: Geoff Holmes

Abstract

We introduce Refinery, an open source platform for exploring large text document collections with topic models. Refinery is a standalone web application driven by a graphical interface, so it is usable by those without machine learning or programming expertise. Users can interactively organize articles by topic and also refine this organization with phrase-level analysis. Under the hood, we train Bayesian nonparametric topic models that can adapt model complexity to the provided data with scalable learning algorithms. The project website `http://daeilkim.github.io/refinery/` contains Python code and further documentation.

Keywords: topic models, visualization, software

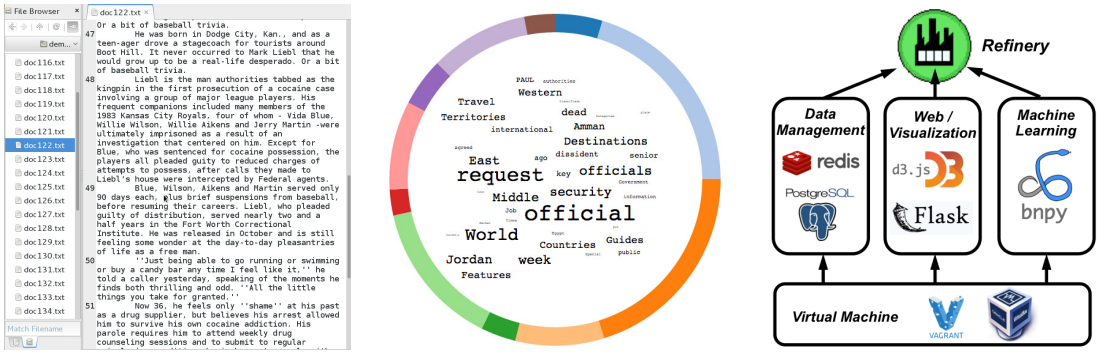
1. Introduction

Topic models have become a ubiquitous class of tools for the analysis of large document collections (Blei, 2012). However, there is still a significant adoption barrier for individuals who have little to no background in coding or computer science. For example, librarians might be interested in understanding ways of organizing their vast archives of information while journalists could potentially use learned topics to extract insights from Freedom of Information Act (FOIA) requests. However, few have the expertise to implement necessary code nor an intuition for how to fine tune parameters associated with these models. Beyond the challenge of running statistical inference algorithms, the learned topics and word associations must be visualized in a way that is accurate but intuitive.

To make topic modeling algorithms and results more accessible, we built an open source web application called Refinery. Refinery allows users to explore a large corpus of documents and further identify a small set of relevant documents for careful study. The simple drag and drop operation of this toolbox allows researchers to focus on the process of quickly understanding their data without being encumbered by the complexities of inference.

2. Running Refinery

Refinery is an in-browser web application that builds on many existing open-source projects related to visualization, data management, and machine learning, as shown in Fig. 1. To make installation simple, it has only three dependencies: the Git version-control system,



(a) *Input:* Plain-text files, one per document

(b) *Output:* Interactive visualization of learned topics

(c) Technology stack

Figure 1: Refinery is a web-application that allows users to upload plain-text documents (a) and visualize the semantic themes learned by a topic-model (b). Under the hood, Refinery uses many existing technologies (c) for browser interaction, data management, machine learning, and visualization. A video demo is available online: `http://youtu.be/7yRQ1J9Z_LI`.

Virtualbox (Oracle, 2013), and Vagrant (Hashimoto, 2013). At a Unix-like command line with these dependencies installed, running Refinery requires just a few lines of code:

```
> git clone https://github.com/daeilkim/refinery.git
> vagrant up # launch machine and install required Python packages
```

After executing these lines, Refinery can be accessed from a web browser via the local address `http://11.11.11.11:8080`.

3. Topic Modeling

Uploading documents. Users submit documents for analysis in the form of a `.zip` or `.tar.gz` file, which uncompressed contains a folder of plain-text files, one for each individual document. Users do *not* need to preprocess documents or identify a vocabulary in advance. This is done automatically by Refinery, which tokenizes every unigram from the input documents. The final vocabulary excludes very rare and very common words; we discard terms appearing in more than 80% of all documents, or in less than 2 documents.

Training a topic model. The topic modeling algorithm used for Refinery comes from the BNPy toolbox (Hughes et al., 2015b). Specifically, we train a *hierarchical Dirichlet process* (HDP) topic model (Teh et al., 2006) using a memoized variational inference algorithm (Hughes et al., 2015a). While the underlying algorithm has many free parameters, we set most of these to smart defaults and only ask the user to specify an upper bound on the number of inferred topics. This bound provides a truncation for infinite-dimensional HDP model.

Browseable visualization. After training a model, users can explore the resulting topics interactively via a multi-colored ring, as shown in Fig. 2. Each segment of the ring represents one topic, with its size indicative of the topic's frequency in the overall corpus. Hovering

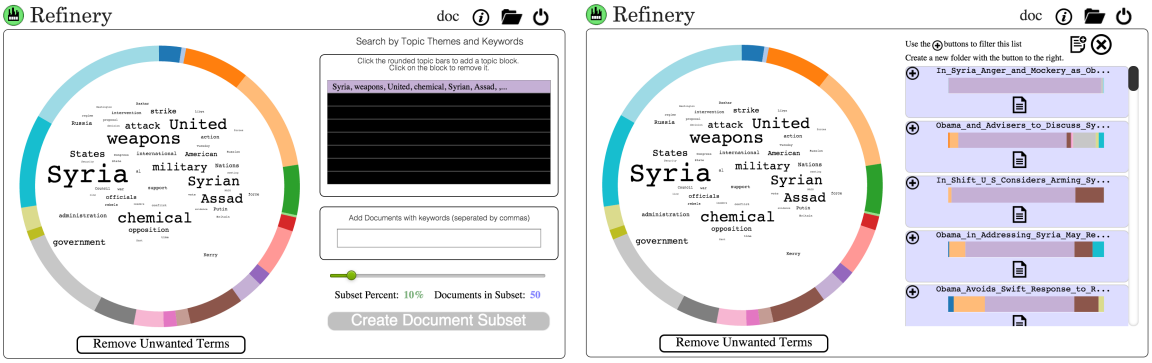


Figure 2: The results of an analysis on 500 New York Times articles that contained the keyword "obama" during the year 2013. In the figure above, the "Syria" topic has been selected along with a subset of 50 documents related to this topic.

the cursor over a topic's segment shows the top 50 words associated with that topic. In this word cloud visualization, text size is scaled according to the topic-specific word frequency.

Focused exploration of document subsets. Often, analysis of large collections requires identifying subsets of relevant documents and performing more detailed clustering of that subset. We support this by allowing the main corpus of documents to be filtered by keyword and topic presence. Refinery provides two methods for selecting relevant documents given a trained topic model with K topics. The first allows the user to directly specify a query distribution over topics which recovered documents should emulate. The second takes in a list of search terms, and averages the probability of topic given word for each term to obtain a (normalized) distribution over K topics. Given this K-dimensional query vector, documents are ranked based on the KL divergence between their MAP topic distribution and the query.

4. Phrase Extraction and Refinement

While topic modeling enables scalable first-pass analysis, often the underlying bag-of-words assumptions are too limiting. After filtering a large corpus to find only documents relevant to a small set of topics, Refinery further allows users to explore documents by phrase similarity and create a summary of the corpus composed of sentences extracted from the input documents.

We use the open-source implementation of the Splitta algorithm (Gillick, 2010), which Gillick (2009) introduced for sentence boundary detection. First, the Splitta algorithm is applied for sentence boundary detection. This algorithm processes raw text and does not make use of the topic model output. Next, to help the user efficiently select a subset of relevant sentences from their corpus, Refinery offers two exploration modes. The first, triggered by the button marked "Variety", implements the KLSum algorithm for multidocument summarization (Haghighi and Vanderwende, 2009). This algorithm ranks candidate sentences higher if their addition to the current set would bring the unigram distribution closer to that of the corpus as a whole, intuitively preferring sentences that contain globally

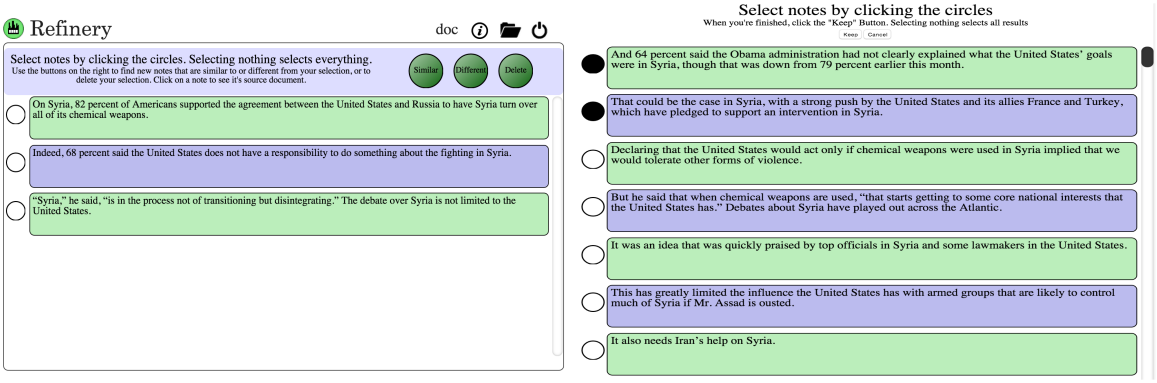


Figure 3: Refinery's phrase extraction feature. *Left:* User has selected some initial phrases related to the war in Syria from news articles. *Right:* Hitting the "similar" button brings up another set of phrases similar to the initial set.

relevant information that does not yet appear in the set of selected sentences. The second exploration method, labeled "Similar", simply ranks candidates by cosine similarity to the current summary's unigram distribution. Each sentence in the summary is linked to and highlighted in its original source document, facilitating the creation of a comprehensive set of notes with fast access to their provenance.

5. Discussion and Related Work

Refinery provides a first step towards allowing non-technical professionals to explore large document collections with modern topic models. Several others groups have strived to simplify and democratize the use of topic models. Notably, Chaney and Blei (2012) created a web-based navigator to help users understand the relationship between topics and documents, while Chuang et al. (2012) developed a visualization package for assessing the goodness of topics. Topicnets (Gretarsson et al., 2012) focuses on a graph-based exploration and visualization of topics, while other packages such as Gephi (Bastian et al., 2009) and Tethne (Peirson et al., 2015) are often used to create similar visualizations of an already-learned set of topics. Most prior work has focused on the parametric *latent Dirichlet allocation* (LDA) topic model (Blei et al., 2003).

Refinery differs from these packages in the way it simplifies the entire process of analyzing text with topic models for non-expert users. Furthermore, Refinery's use of scalable Bayesian nonparametric topic models offers the benefit of automatic model selection while parametric models like LDA require a prior knowledge of the number of topics. Refinery also supports phrase extraction, which allows for a more refined search across documents. Building on the BNPy toolbox allows potential future extensions to more advanced models that cluster documents organized as a time-series or network. Ultimately, we plan to support a larger variety of potential document file types including structured word processor files, spreadsheets, and presentations.

Acknowledgments

Refinery was a recipient of the Knight Prototype Fund in 2014.

References

Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: An open source software for exploring and manipulating networks. In *International AAAI Conference on Weblogs and Social Media*, 2009. URL `http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154`.

David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

Allison June-Barlow Chaney and David M Blei. Visualizing topic models. In *ICWSM*, 2012.

Jason Chuang, Christopher D. Manning, and Jeffrey Heer. Termite: Visualization techniques for assessing textual topic models. In *Advanced Visual Interfaces*, 2012. URL `http://vis.stanford.edu/papers/termite`.

Dan Gillick. Sentence boundary detection and the problem with the U.S. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 241–244. Association for Computational Linguistics, 2009.

Daniel Gillick. splitta: statistical sentence boundary detection, 2010. URL `https://code.google.com/p/splitta/`.

Brynjar Gretarsson, John Odonovan, Svetlin Bostandjiev, Tobias Höllerer, Arthur Asuncion, David Newman, and Padhraic Smyth. Topicnets: Visual analysis of large text corpora with topic modeling. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(2):23, 2012.

Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370. Association for Computational Linguistics, 2009.

Mitchell Hashimoto. *Vagrant: Up and Running*. O'Reilly Media, Inc., 2013. URL `http://www.vagrantup.com`.

Michael C. Hughes, Daeil Kim, and Erik B. Sudderth. Reliable and scalable variational inference for the hierarchical dirichlet process. In *AISTATS*, 2015a.

Michael C. Hughes et al. BNPy: Bayesian nonparametric machine learning for Python, 2015b. URL `https://github.com/bnpy/bnpy/`.

Oracle. Virtualbox, 2013. URL `http://www.virtualbox.org`. Version 4.2.18.

Erick Peirson, Aaron Baker, and Ramki Subramanian. Tethne: Bibliographic network analysis in Python, 2015. URL `https://github.com/diging/tethne`.

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *JASA*, 101 (476):1566–1581, 2006.