OCTIS: Comparing and Optimizing Topic Models is Simple!

Silvia Terragni
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
s.terragni4@campus.unimib.it

Elisabetta Fersini*
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
elisabetta.fersini@unimib.it

Bruno Galuzzi
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
bruno.galuzzi@unimib.it

Pietro Tropeano
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
p.tropeano1@campus.unimib.it

Antonio Candelieri
University of Milano-Bicocca
Viale Sarca 336, 20126
Milan, Italy
antonio.candelieri@unimib.it

Abstract

In this paper, we present OCTIS, a framework for training, analyzing, and comparing Topic Models, whose optimal hyper-parameters are estimated using a Bayesian Optimization approach. The proposed solution integrates several state-of-the-art topic models and evaluation metrics. These metrics can be targeted as objective by the underlying optimization procedure to determine the best hyper-parameter configuration. OCTIS allows researchers and practitioners to have a fair comparison between topic models of interest, using several benchmark datasets and well-known evaluation metrics, to integrate novel algorithms, and to have an interactive visualization of the results for understanding the behavior of each model. The code is available at the following link: https://github.com/MIND-Lab/OCTIS.

1 Introduction

Topic models are promising statistical methods that aim to extract the hidden topics underlying a collection of documents. Although researchers have proposed several models across the years (Blei, 2012; Vayansky and Kumar, 2020), their evaluation and comparison is still a hard task. The evaluation of a topic model usually involves different datasets (with non-standard pre-processing) (Schofield and Mimno, 2016; Schofield et al., 2017) and several evaluation metrics (Lau et al., 2014; Wallach et al., 2009; Terragni et al., 2020a). Furthermore, topic models are usually compared by fixing their hyper-parameters. However, choosing the optimal hyper-parameter configuration for a given dataset and a given evaluation metric is fundamental to induce each model at the best of its capabilities, and therefore to guarantee a fair comparison with other models.

Current topic modeling frameworks (McCallum et al., 2005; Qiang et al., 2018; Lisena et al., 2020) typically focus on the release of topic modeling algorithms while ignoring one or more critical aspects of the topic modeling pipeline, such as preprocessing, evaluation, comparison of the models, and visualization. Most importantly, they disregard the hyper-parameter selection.

In this paper, we present OCTIS (Optimizing and Comparing Topic models Is Simple)[1], a unified and open-source evaluation framework for training, analyzing, and comparing Topic Models, over several datasets and evaluation metrics. Their optimal hyper-parameter configuration is determined according to a Bayesian Optimization (BO) strategy (Archetti and Candelieri, 2019; Snoek et al., 2012; Galuzzi et al., 2020).

In the following, we summarize the main contributions of the proposed framework:

- several open-source topic models have been integrated into a *unified framework*, providing a common interface that allows the users to easily experiment with topic models;
- a single-objective BO approach has been integrated to determine the *optimal hyper-parameter* values of each model, for a given dataset and a specific evaluation metric of interest;
- an *interactive visualization* of the results for inspecting the details of the models, providing

* Corresponding author.

[1] The video demonstration is available at https://youtu.be/nPmiWBFFJ8E.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, pages 263–270
April 19 - 23, 2021.

insights about the optimization strategy, word and topic distributions, and robustness of the estimated configuration;

- a *python library* for advanced exploitation of the framework for integrating novel algorithms, with their training and inference algorithms.

2 System design and architecture

OCTIS is an open-source evaluation framework for the comparison of a set of state-of-the-art topic models, that allows the user to optimize the models' hyper-parameters for a fair experimental comparison. The proposed framework follows an object-oriented paradigm, providing all the tools for running a topic modeling pipeline.

The main functionalities of the proposed OCTIS are related to dataset pre-processing, training topic models, estimating evaluation metrics, hyper-parameter optimization, and interactive web dashboard visualization. Figure 1 summarizes the workflow involving the first four modules (the dashboard interacts with all of them).

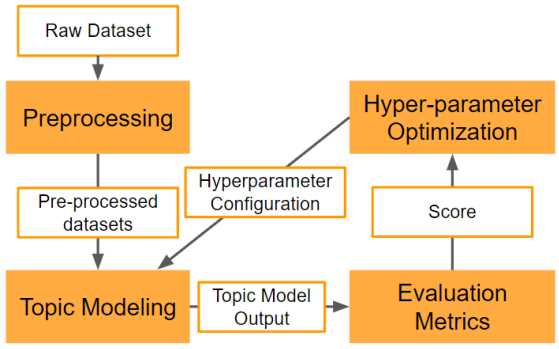

Figure 1: Workflow of the OCTIS framework

The framework can be used both as a python library and as a dashboard. The python library offers more advanced functionalities than the ones available in the dashboard. The modules that comprise the OCTIS framework are detailed in the following sections.

2.1 Datasets and Pre-processing

The first step of the topic modeling pipeline is the pre-processing of the input dataset. OCTIS includes the following pre-processing utilities:

- reducing the text to lowercase;
- punctuation removal;
- lemmatization;
- stop-words removal;
- removal of unfrequent and most frequent words (according to a specified frequency threshold);
- removal of documents with few words (according to a specified frequency threshold).

These utilities include the most common techniques for pre-processing text for topic modeling. However, some of these features may not be appropriate for specific domains and languages, e.g. requiring language-specific or domain-specific stop-words.

OCTIS currently provides 4 pre-processed datasets, i.e. 20 NewsGroups [2], M10 (Lim and Buntine, 2014), DBLP [3] and BBC News (Greene and Cunningham, 2006), different in nature and length.

The datasets already available in OCTIS, and accessible through the web dashboard, have been pre-processed according to the length of the documents. In particular, we removed the punctuation, we lemmatized the text, filtered out the stop-words (using the English stop-words list provided by MALLET), and removed the words that have a word frequency less than 0.5% for 20 Newsgroups and BBC News and less than 0.05% for DBLP and M10. Subsequently, we removed the documents with less than 5 words for 20 Newsgroups and BBC News and less than 3 words for the other datasets.

Table 1 reports some statistics about the currently available pre-processed.

Dataset	Domain	# Docs	Avg # words in docs	# Unique words
20 Newsgroups	Forum posts	16309	48	1612
BBC News	News	2225	150	3106
M10	Scientific papers	8355	6	1696
DBLP	Scientific papers	54595	5	1513

Table 1: Statistics of the pre-processed datasets.

[2] http://people.csail.mit.edu/jrennie/20Newsgroups/

[3] https://github.com/shiruipan/TriDNR/tree/master/data

Although OCTIS already provides some datasets, a user can upload and pre-process any dataset (using the python library) according to its needs.

2.2 Topic Modeling

OCTIS integrates both classical topic models and neural topic models. In particular, the following traditional and neural approaches are available to be trained, optimized, analyzed, and compared (the models that are available in the web dashboard are marked with ⋆):

- Latent Dirichlet Allocation⋆ (Blei et al., 2003, LDA);[4]
- Non-negative Matrix Factorization⋆ (Lee and Seung, 2000, NMF);[4]
- Latent Semantic Analysis⋆ (Hofmann, 1999, LSI);[4]
- Hierarchical Dirichlet Process (Teh et al., 2004, HDP);[4]
- Neural LDA⋆ (Srivastava and Sutton, 2017);[5]
- Product-of-Experts LDA⋆ (Srivastava and Sutton, 2017, ProdLDA);[5]
- Embedded Topic Models⋆ (Dieng et al., 2019, ETM);[6]
- Contextualized Topic Models (Bianchi et al., 2021, CTM).[7]

Moreover, we defined a standard interface for allowing a user to integrate their topic model's implementation. A topic model is indeed a black-box, a system solely viewed in terms of its inputs and outputs and whose internal workings are invisible. This black-box topic model takes as input a dataset and a set of hyperparameters values and returns the top-t topic words, the document-topic distributions, and the topic-word distribution in a specified format.

[4] https://radimrehurek.com/gensim/
[5] https://github.com/estebandito22/PyTorchAVITM
[6] https://github.com/adjidieng/ETM
[7] https://github.com/MilaNLProc/contextualized-topic-models

2.3 Evaluation Metrics

The proposed framework provides several evaluation metrics. A metric can be used as the objective targeted by a Bayesian Optimization strategy, or to monitor the behavior of a topic model while the model is optimized on a different objective. The performance of a topic model can be evaluated by investigating different aspects, according to the following evaluation metrics:

- Topic coherence metrics (Lau et al., 2014; Röder et al., 2015) that compute how the top-k words of a topic are related to each other;
- Topic significance metrics (AlSumait et al., 2009; Terragni et al., 2020b) that focus on the document-topic and topic-word distributions to discover high-quality and junk topics;
- Diversity metrics (Dieng et al., 2019; Bianchi et al., 2020) that measure how diverse the top-k words of a topic are to each other;
- Classification metrics (Phan et al., 2008; Terragni et al., 2020a) where the document-topic distribution of each document is used as the K-dimensional representation to train a classifier that predicts the document's class.

OCTIS provides 10 evaluation metrics directly available in the web dashboard, and 13 accessible through the python library.

2.4 Hyper-parameter Optimization

The proposed framework uses Bayesian Optimization (Snoek et al., 2012; Shahriari et al., 2015) to tune the hyper-parameters of the topic models. If any of the available hyper-parameters is selected to be optimized for a given evaluation metric, BO explores the search space to determine the optimal settings. Since the performance estimated by the evaluation metrics can be affected by noise, the objective function is computed as the median of a given number of *model runs* (i.e., topic models run with the same hyperparameter configuration) computed for the selected evaluation metric.

BO is a sequential model-based optimization strategy for expensive and noisy black-box functions (e.g. topic models). The basic idea consists of using all the model's configurations evaluated so far to approximate the value of the performance metric and then selects a new promising configuration to evaluate.

Features	OCTIS	Gensim	STTM	PyCARET	MALLET	TOMODAPI
Pre-processing tools	√	√		√	√	√
Pre-processed datasets	√	√	√	√		√
Classical topic models	√	√	√	√	√	√
Neural topic models	√					√
Coherence metrics	√	√	√	√	√	
Diversity metrics	√					
Significance metrics	√					
Classification metrics	√		√	√	√	√
Hyper-parameters tuning	BO	MLE		grid-search	MLE	
Usage	import in script, web dashboard	import in script	command line	import in script	command line	import in script, API
Programming Language	Python	Python	Java	Python	Java	Python

Table 2: Comparison between OCTIS and the most well-known topic modeling libraries.

The approximation is provided by a probabilistic *surrogate model*, which describes the prior belief over the objective function using the observed configurations. The next configuration to evaluate is selected through the optimization of an *acquisition function*, which leverages the uncertainty in the posterior to guide the exploration.

We integrated into OCTIS most of the BO algorithms of the Scikit-Optimize library (Head et al., 2018) to provide a robust and efficient BO implementation. We integrated Gaussian Process and Random Forest as surrogate models, while we included Probability of Improvement, Expected Improvement, and Upper Confidence Bound as acquisition functions. See (Snoek et al., 2012; Candelieri and Archetti, 2019) for more details about the use of BO for hyper-parameter optimization.

Instead of performing BO, a user can also use a random search technique to find the best hyper-parameter configuration. Since the Bayesian Optimization requires some initial configurations to fit the surrogate model, the user can provide the initial configurations, according to their domain knowledge. Alternatively, a user can perform a pure exploration of the search space using a random sampling strategy. Different algorithms are available (e.g. Uniform Random Sampling or Latin Hypercube sequence) for sampling the initial configurations.

3 Existing frameworks

The existing topic modeling frameworks usually provide topic modeling algorithms, while disregarding other essential aspects of the whole topic modeling pipeline: pre-processing, evaluation, comparison, and visualization of the results and, most importantly, the hyper-parameter selection. In the following, we outline the existing frameworks, highlighting their advantages and limitations.

MALLET(McCallum, 2002) and gensim[4] are the most known topic modeling libraries and include several classical topic models. They provide pre-processing methods and the estimation of the hyper-parameters using maximum likelihood estimation (MLE) techniques. These libraries do not include the recently proposed neural topic models, and they just provide topic coherence metrics.

STTM (Qiang et al., 2018) is a java library that provides a set of topic models that are specifically designed for short texts, providing several evaluation metrics.

ToModAPI (Lisena et al., 2020) is a python API that allows for training, inference, and evaluating different topic models, also including some of the most recent. However, it does not provide a method for finding the best hyper-parameter configuration of topic models. Instead, a tool that allows for optimizing the hyper-parameter of a machine learning model is PyCARET (Ali, 2020). However, it employs a grid-search technique to tune the hyper-parameters. This approach can be very time-consuming if the number of hyperparameters is high and the search space is huge (Bergstra and Bengio, 2012).

OCTIS stands at the union of the features of the existing frameworks: we integrated both classical and recent neural topic models, providing pre-processing methods, evaluation metrics, and the possibility of optimizing the hyper-parameters. Finally, a user-friendly graphical interface to launch one or more hyper-parameter optimization experiments on a given topic model and on a specific dataset has been provided.

Table 2 summarizes the main features of the

existing topic modeling frameworks and compares them with OCTIS.

4 System usage

OCTIS has been designed to be used as a python library by advanced users, as well as a simple web dashboard by anyone.

4.1 Example of use case for the python library

```
# loading of a pre-processed dataset
dataset = Dataset()
dataset.load("path/to/dataset")

#model instantiation
lda = LDA(num_topics=25)

#definition of the metric
td = TopicDiversity()

#definition of the search space
search_space = {
  "eta": Real(low=0.01, high=5.0),
  "alpha": Real(low=0.01, high=5.0)
}

#define and launch optimization
optimizer=Optimizer()
opt_result = optimizer.optimize(model,
     dataset, td, search_space)
```

The above lines of code will execute an optimization experiment that will provide an optimal configuration of the hyperparameters α and β for LDA with 25 topics by maximizing the diversity of the topics.

4.2 Web-based dashboard

The dashboard includes a set of simple but useful operations to conduct an experimental campaign on different topic models. Here we briefly explain the four main functionalities of the dashboard.

Experiment creation. First, a user can define an optimization experiment by selecting the dataset, the topic model, the corresponding hyperparameter to optimize, the evaluation metric to be considered by the BO (possibly other extra metrics to evaluate), and the settings of the optimization process.

Management of the experiments' queue. The user can monitor the queue of the experiments and see the corresponding progress. The user can also pause, restart, or delete an experiment that has been launched before. Additionally, the user can easily change the order of the queue of the experiments, by allowing a given run to be executed before others.

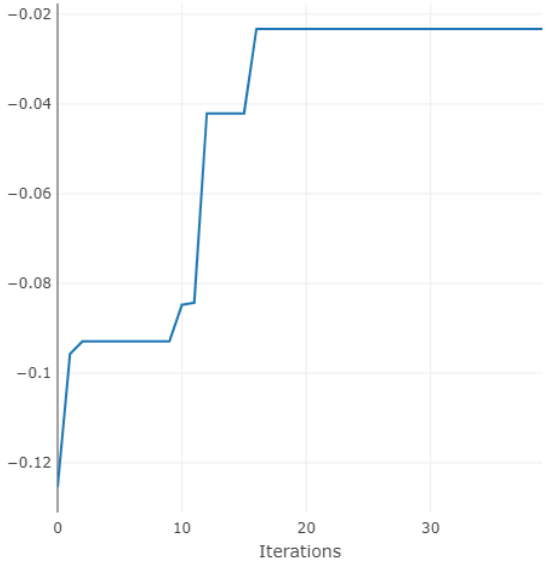

Figure 2: Example of the best-seen evolution for an optimization experiment.

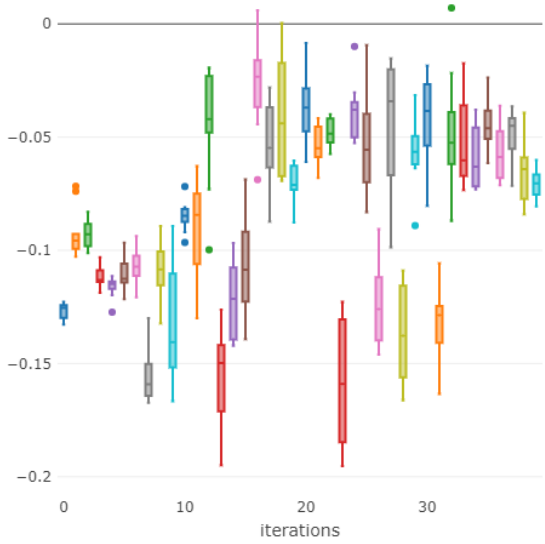

Figure 3: Example of box plot of an optimization experiment.

Comparison of the Topic Models. The user can select the models to be analyzed and compared. At the first stage, one can observe the progress of the BO iterations, observing a plot that contains at each iteration the best-seen evaluation, i.e. the median at each iteration of the metric that has been optimized (see Figure 2). Alternatively, a user can visualize a

box plot at each iteration (see Figure 3) to understand if a given hyper-parameter configuration is noisy (high variance) or not.

Analysis of a single experiment. A user can further inspect the results of a specific topic model on a given dataset with respect to the considered metrics, by analyzing a single experiment.

Here, a user can visualize all the information and statistics related to the experiment, including the best hyper-parameter configuration and the best value of the optimized metric. They can also have an outline of the statistics of the other extra metrics that they had chosen to evaluate.

Figure 4: Example of word cloud of a topic.

We provide three different plots for inspecting the output of a single run of a topic model. Figure 4 shows the word cloud obtained from the most relevant words of a given topic, scaled by their probability. Focusing on the distributions inferred by a topic model, Figure 5 shows the topic distribution of a document, and Figure 6 represents an example of the weight of a selected word of the vocabulary for each topic.

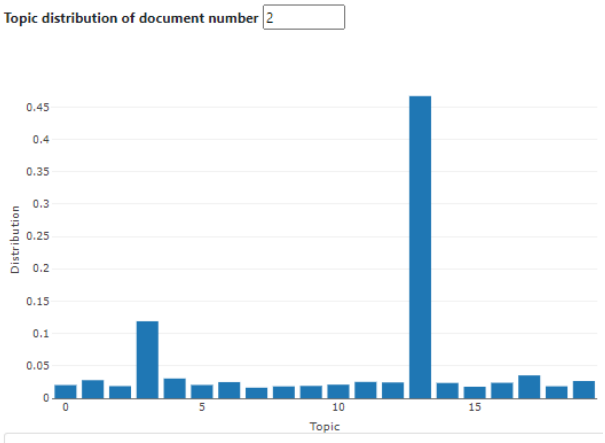

Figure 5: Example of distribution of the topics in a selected document.

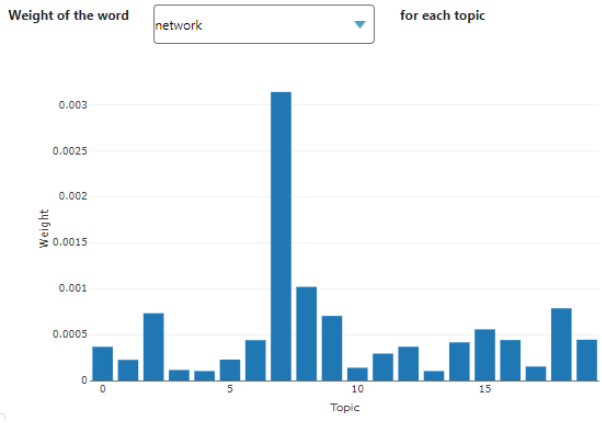

Figure 6: Example of the weight of the word "network" for each document.

5 Conclusions

In this paper, we presented the framework OCTIS for training, analyzing, and comparing Topic Models. The proposed framework is composed of a python library and a web dashboard and integrates several state-of-the-art topic models (both traditional and neural). These models can be trained by searching for their optimal hyperparameter configuration, for a given metric and dataset, exploiting a BO strategy. OCTIS allows researchers to train existing models, integrate new training and inference algorithms, and fairly compare the topic models of interest. On the other hand, practitioners could use OCTIS to boost the performance of Topic Models for their preferred downstream task or a wide range of practical applications, such as data exploratory analysis (Boyd-Graber et al., 2017).

Regarding future work, OCTIS could integrate a multi-objective optimization strategy to optimize multiple metrics in the same BO procedure (Paria et al., 2020). For example, this could allow a user to find an optimal hyper-parameter configuration for both topic coherence and document classification.

References

Moez Ali. 2020. *PyCaret: An open source, low-code machine learning library in Python*. PyCaret version 2.3.

Loulwah AlSumait, Daniel Barbará, James Gentle, and Carlotta Domeniconi. 2009. Topic significance ranking of LDA generative models. In *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2009*, volume 5781 of *Lecture Notes in Computer Science*, pages 67–82. Springer.

Francesco Archetti and Antonio Candelieri. 2019. *Bayesian Optimization and Data Science*. Springer International Publishing.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2020. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. *arXiv preprint arXiv:2004.03974*.

Federico Bianchi, Silvia Terragni, Dirk Hovy, Debora Nozza, and Elisabetta Fersini. 2021. Cross-lingual Contextualized Topic Models with Zero-shot Learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*.

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

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jordan L. Boyd-Graber, Yuening Hu, and David M. Mimno. 2017. Applications of topic models. *Found. Trends Inf. Retr.*, 11(2-3):143–296.

Antonio Candelieri and Francesco Archetti. 2019. Global optimization in machine learning: the design of a predictive analytics application. *Soft Comput.*, 23(9):2969–2977.

Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. 2019. Topic modeling in embedding spaces. *CoRR*, abs/1907.04907.

BG Galuzzi, I Giordani, A Candelieri, R Perego, and F Archetti. 2020. Hyperparameter optimization for recommender systems through bayesian optimization. *Computational Management Science*, pages 1–21.

Derek Greene and Pádraig Cunningham. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd International Conference on Machine learning (ICML'06)*, pages 377–384. ACM Press.

Tim Head, Gilles Louppe MechCoder, Iaroslav Shcherbatyi, et al. 2018. scikit-optimize/scikit-optimize: v0. 5.2.

Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57. ACM.

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014*, pages 530–539.

Daniel D. Lee and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000*, pages 556–562. MIT Press.

Kar Wai Lim and Wray L. Buntine. 2014. Bibliographic analysis with the citation network topic model. In *Proceedings of the Sixth Asian Conference on Machine Learning, ACML 2014*.

Pasquale Lisena, Ismail Harrando, Oussama Kandakji, and Raphael Troncy. 2020. ToModAPI: A Topic Modeling API to Train, Use and Compare Topic Models. In *2^{nd} International Workshop for Natural Language Processing Open Source Software (NLP-OSS)*.

Andrew McCallum, Andrés Corrada-Emmanuel, and Xuerui Wang. 2005. Topic and role discovery in social networks. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 786–791.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. *http://mallet. cs. umass. edu*.

Biswajit Paria, Kirthevasan Kandasamy, and Barnabás Póczos. 2020. A flexible framework for multi-objective bayesian optimization using random scalarizations. In *Uncertainty in Artificial Intelligence*, pages 766–776. PMLR.

Xuan Hieu Phan, Minh Le Nguyen, and Susumu Horiguchi. 2008. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web, WWW 2008*, pages 91–100. ACM.

Jipeng Qiang, Yun Li, Yunhao Yuan, Wei Liu, and Xindong Wu. 2018. Sttm: A tool for short text topic modeling. *arXiv preprint arXiv:1808.02215*.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, Shanghai, China, February 2-6, 2015*, pages 399–408. ACM.

Alexandra Schofield, Måns Magnusson, and David Mimno. 2017. Pulling out the stops: Rethinking stopword removal for topic models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 432–436.

Alexandra Schofield and David Mimno. 2016. Comparing apples to apple: The effects of stemmers on topic models. *Transactions of the Association for Computational Linguistics*, 4:287–300.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. 2015. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012*, pages 2960–2968.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. In *5th International Conference on Learning Representations, ICLR 2017*.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2004. Sharing clusters among related groups: Hierarchical dirichlet processes. In *Advances in Neural Information Processing Systems, 17*, pages 1385–1392.

Silvia Terragni, Elisabetta Fersini, and Enza Messina. 2020a. Constrained relational topic models. *Information Sciences*, 512:581 – 594.

Silvia Terragni, Debora Nozza, Elisabetta Fersini, and Messina Enza. 2020b. Which matters most? comparing the impact of concept and document relationships in topic models. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 32–40.

Ike Vayansky and Sathish A. P. Kumar. 2020. A review of topic modeling methods. *Information Systems*, 94:101582.

Hanna M Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. 2009. Evaluation methods for topic models. In *Proceedings of the 26th annual international conference on machine learning*, pages 1105–1112.