ToModAPI: A Topic Modeling API to Train, Use and Compare Topic Models

Pasquale Lisena, Ismail Harrando, Oussama Kandakji and Raphaël Troncy
EURECOM, Sophia Antipolis, France
{firstname.lastname}@eurecom.fr

Abstract

From LDA to neural models, different topic modeling approaches have been proposed in the literature. However, their suitability and performance is not easy to compare, particularly when the algorithms are being used in the wild on heterogeneous datasets. In this paper, we introduce ToModAPI (*TOpic MOdeling API*), a wrapper library to easily train, evaluate and infer using different topic modeling algorithms through a unified interface. The library is extensible and can be used in Python environments or through a Web API.

1 Introduction

The analysis of massive volumes of text is an extremely expensive activity when it relies on not-scalable manual approaches or crowdsourcing strategies. Relevant tasks typically include textual document classification, document clustering, keywords and named entities extraction, language or sequence modeling, etc. In the literature, topic modeling and topic extraction, which enable to automatically recognise the main subject (or topic) in a text, have attracted a lot of interest. The predicted topics can be used for clustering documents, for improving named entity extraction (Newman et al., 2006), and for automatic recommendation of related documents (Luostarinen and Kohonen, 2013).

Several topic modeling algorithms have been proposed. However, we argue that it is hard to compare and to choose the most appropriate one given a particular goal. Furthermore, the algorithms are often evaluated on different datasets and different scoring metrics are used. In this work, we have selected some of the most popular topic modeling algorithms from the state of the art in order to integrate them in a common platform, which homogenises the interface methods and the evaluation metrics. The result is ToModAPI[1] which allows to dynamically train, evaluate, perform inference on different models, and extract information from these models as well, making it possible to compare them using different metrics.

The remaining of this paper is organised as follows. In Section 2, we describe some related works and we detail some state-of-the-art topic modeling techniques. In Section 3, we provide an overview of the evaluation metrics usually used. We introduce ToModAPI in Section 4. We then describe some datasets (Section 5) that are used in training to perform a comparison of the topic models (Section 6). Finally, we give some conclusions and outline future work in Section 7.

2 Related Work

Aside from a few exceptions (Blei and McAuliffe, 2007), most topic modeling works propose or apply unsupervised methods. Instead of learning the mapping to a pre-defined set of topics (or labels), the goal of these methods consists in assigning training documents to N unknown topics, where N is a required parameter. Usually, these models compute two distributions: a Document-Topic distribution which represents the probability of each document to belong to each topic, and a Topic-Word distribution which represents the probability of each topic to be represented by each word present in the documents. These distributions are used to predict (or infer) the topic of unseen documents.

Latent Dirichlet Allocation (LDA) is a unsupervised statistical modeling approach (Blei et al., 2003) that considers each document as a *bag of words* and creates a randomly assigned document-topic and word-topic distribution. Iterating over words in each document, the distributions are updated according to the probability that a document

[1] ToModAPI: TOpic MODeling API

Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS), pages 132–140
Virtual Conference, November 19, 2020.

or a word belongs to a certain topic. The **Hierarchical Dirichlet Process (HDP)** model (Teh et al., 2006) is another statistical approach for clustering grouped data such as text documents. It considers each document as a group of words belonging with a certain probability to one or multiple components of a mixture model, i.e. the topics. Both the probability measure for each document (distribution over the topics) and the base probability measure – which allows the sharing of clusters across documents – are drawn from Dirichlet Processes (Ferguson, 1973). Differently from many other topic models, HDP infers the number of topics automatically.

Gibbs Sampling for a DMM (GSDMM) applies the Dirichlet Multinomial Mixture model for short text clustering (Yin and Wang, 2014). This algorithm works computing iteratively the probability that a document join a specific one of the N available clusters. This probability consist in two parts: 1) a part that promotes the clusters with more documents; 2) a part that advantages the movement of a document towards similar clusters, i.e. which contains a similar word-set. Those two parts are controlled by the parameters α and β. The simplicity of GSDMM provides a fast convergence after some iterations. This algorithm consider the given number of clusters given as an upper bound and it might end up with a lower number of topics. From another perspective, it is somehow able to infer the optimal number of topics, given the upper bound.

Pre-trained Word vectors such as word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014) can help to enhance topic-word representations, as achieved by the **Latent Feature Topic Models (LFTM)** (Nguyen et al., 2015). One of the LFTM algorithms is *Latent Feature LDA (LF-LDA)*, which extends the original LDA algorithm by enriching the topic-word distribution with a latent feature component composed of pre-trained word vectors. In the same vein, the **Paragraph Vector Topic Model (PVTM)** (Lenz and Winker, 2020) uses doc2vec (Le and Mikolov, 2014) to generate document-level representations in a common embedding space. Then, it fits a Gaussian Mixture Model to cluster all the similar documents into a predetermined number of topics – i.e. the number of GMM components.

Topic modeling can also be performed via linear-algebraic methods. Starting from the the high-dimensional term-document matrix, multiple approaches can be used to lower its dimensions. Then, we consider every dimension in the lower-rank matrix as a latent topic. A straightforward application of this principle is the **Latent Semantic Indexing model (LSI)** (Deerwester et al., 1990), which uses Singular Value Decomposition as a means to approximate the term-document matrix (potentially mediated by TF-IDF) into one with less rows – each one representing a latent semantic dimension in the data – and preserving the similarity structure among columns (terms). **Non-negative Matrix Factorisation (NMF)** (Paatero and Tapper, 1994) exploits the fact that the term-document matrix is non-negative, thus producing not only a denser representation of the term-document distribution through the matrix factorisation but guaranteeing that the membership of a document to each topic is represented by a positive coefficient.

In recent years, neural network approaches for topic modeling have gained popularity giving birth to a family of **Neural Topic Models (NTM)** (Cao et al., 2015). Among those, **doc2topic (D2T)**[2] uses a neural network which separately computes N-dimensional embedding vectors for words and documents – with N equal to the number of topics, before computing the final output using a sigmoid activation. The distributions topic-word and document-topic are obtained by getting the final weights on the two embedding layers. Another neural topic model, the **Contextualized Topic Model (CTM)** (Bianchi et al., 2020) uses Sentence-BERT (SBERT) (Reimers and Gurevych, 2019) – a neural transformer language model designed to compute sentences representations efficiently – to generate a fixed-size embedding for each document to contextualise the usual Bag of Words representation. CTM enhances the *Neural-ProdLDA* (Srivastava and Sutton, 2017) architecture with this contextual representation to significantly improve the coherence of the generated topics.

Previous works have tried to compare different topic models. A review of statistical topic modeling techniques is included in Newman et al. (2006). A comparison and evaluation of LDA and NMF using the coherence metric is proposed by O'Callaghan et al. (2015). Among the libraries for performing topic modeling, *Gensim* is undoubtedly the most known one, providing implementations of

[2] https://github.com/sronnqvist/doc2topic

several tools for the NLP field (Řehůřek and Sojka, 2010). Focusing on topic modeling for short texts, *STMM* includes 11 different topic models, which can be trained and evaluated through command line (Qiang et al., 2019). The *Topic Modelling Open Source Tool*[3] exposes a web graphical user interface for training and evaluating topic models, LDA being the only representative so far. The *Promoss Topic Modelling Toolbox*[4] provides a unified Java command line interface for computing a topic model distribution using LDA or the *Hierarchical Multi-Dirichlet Process Topic Model (HMDP)* (Kling, 2016). However, it does not allow to apply the computed model on unseen documents.

3 Metrics

The evaluation of machine learning techniques often relies on accuracy scores computed comparing predicted results against a ground truth. In the case of unsupervised techniques like topic modeling, the ground truth is not always available. For this reason, in the literature, we can find:

- metrics which enable to evaluate a topic model independently from a ground truth, among which, coherence measures are the most popular ones for topic modeling (Röder et al., 2015; O'Callaghan et al., 2015; Qiang et al., 2019);
- metrics that measure the quality of a model's predictions by comparing its resulting clusters against ground truth labels, in this case a topic label for each document.

3.1 Coherence metrics

The coherence metrics rely on the joint probability $P(w_i, w_j)$ of two words w_i and w_j that is computed by counting the number of documents in which those words occur together divided by the total number of documents in the corpus. The documents are fragmented using sliding windows of a given length, and the probability is given by the number of fragments including both w_i and w_j divided by the total number of fragments. This probability can be expressed through the *Pointwise Mutual Information (PMI)*, defined as:

$$PMI(w_i, w_j) = log\frac{P(w_i, w_j) + \epsilon}{P(w_i) \cdot P(w_j)} \qquad (1)$$

A small value is chosen for ϵ, in order to avoid computing the logarithm of 0. Different metrics based on PMI have been introduced in the literature, differing in the strategies applied for token segmentation, probability estimation, confirmation measure, and aggregation. The **UCI coherence** (Röder et al., 2015) averages the PMI computed between pairs of topics, according to:

$$C_{UCI} = \frac{2}{N\cdot(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} PMI(w_i, w_j) \quad (2)$$

The **UMASS coherence** (Röder et al., 2015) relies instead on a differently computed joint probability:

$$C_{UMASS} = \frac{2}{N\cdot(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} log\frac{P(w_i,w_j)+\epsilon}{P(w_j)} \quad (3)$$

The **Normalized Pointwise Mutual Information (NPMI)** (Chiarcos et al., 2009) applies the PMI in a confirmation measure for defining the association between two words:

$$NPMI(w_i, w_j) = \frac{PMI(w_i, w_j)}{-log(P(w_i, w_j) + \epsilon)} \quad (4)$$

NPMI values go from -1 (never co-occurring words) to +1 (always co-occurring), while the value of 0 suggests complete independence. This measure can be applied also to word sets. This is made possible using a vector representation in which each feature consists in the NPMI computed between w_i and a word in the corpus W, according to the formula:

$$\vec{v}(w_i) = \left\{ NPMI(w_i, w_j) | w_j \in W \right\} \quad (5)$$

In ToModAPI, we include the following four metrics[5]:

- C_{NPMI} applies NPMI as in Eqn (4) to couples of words, computing their joint probabilities using sliding windows;
- C_V compute the cosine similarity of the vectors – as defined in Eqn (5) – related to each word of the topic. The NPMI is computed on sliding windows;
- C_{UCI} as in Eqn (2);
- C_{UMASS} as in Eqn (3).

[3] https://github.com/opeyemibami/Topic-Modelling-Open-Source-Tool

[4] https://github.com/gesiscss/promoss

[5] We use the implementation of these metrics as provided in Gensim. The window size is kept at the default values.

Additionally, we include a **Word Embeddings-based Coherence** as introduced by Fang et al. (2016). This metric relies on pre-trained word embeddings such as GloVe or word2vec and evaluate the topic quality using a similarity metric between its top words. In other words, a high mutual embedding similarity between a model's top words reflects its underlying semantic coherence. In the context of this paper, we will use the sum of mutual cosine similarity computed on the Glove vectors[6] of the top $N = 10$ words of each topic:

$$C_{WE} = \frac{2}{N \cdot (N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} cos(v_i, v_j) \quad (6)$$

where v_i and v_j are the GloVe vectors of the words w_i and w_j.

All metrics aggregate the different values at topic level using the arithmetic mean, in order to provide a coherence value for the whole model.

3.2 Metrics which relies on a ground truth

The most used metric that relies on a ground truth is the **Purity**, defined as the fraction of documents in each cluster with a correct prediction (Hajjem and Latiri, 2017). A prediction is considered correct if the original label coincides with the original label of the majority of documents falling in the same topic prediction. Given L the set of original labels and T the set of predictions:

$$Purity(T, L) = \frac{1}{|T|} \sum_{i \in T} \max_{j \in L} |T_j \cap L_j| \quad (7)$$

In addition, we include in the API the following metrics used in the literature for evaluating the quality of classification or clustering algorithms, applied to the topic modeling task:

1. **Homogeneity**: a topic model output is considered homogeneous if all documents assigned to each topic belong to the same ground-truth label (Rosenberg and Hirschberg, 2007);
2. **Completeness**: a topic model output is considered complete if all documents from one ground-truth label fall into the same topic (Rosenberg and Hirschberg, 2007);
3. **V-Measure**: the harmonic mean of Homogeneity and Completeness. A V-Measure of 1.0 corresponds to a perfect alignment between topic model outputs and ground truth labels (Rosenberg and Hirschberg, 2007);
4. **Normalized Mutual Information (NMI)** is the ratio between the mutual information between two distributions – in our case, the prediction set and the ground truth – normalised through an aggregation of those distributions' entropies (Lancichinetti et al., 2009). The aggregation can be realised by selecting the minimum/maximum or applying the geometric/arithmetic mean. In the case of arithmetic mean, NMI is equivalent to the V-Measure.

For these metrics, we use the implementations provided by scikit-learn (Pedregosa et al., 2011).

4 ToModAPI: a Topic Modeling API

We now introduce ToModAPI, a Python library which harmonises the interfaces of topic modeling algorithms. So far, 9 topic modeling algorithms have been integrated in the library (Table 1).

For each algorithm, the following interface methods are exposed:

- `train` which requires in input the path of a dataset and an algorithm-specific set of training parameters;
- `topics` which returns the list of trained topics and, for each of them, the 10 most representative words. Where available, the weights of those words in representing the topic are given;
- `topic` which returns the information (representative words and weights) about a single topic;
- `predict` which performs the topic inference on a given (unseen) text;
- `get_training_predictions` which provides the final predictions made on the training corpus. Where possible, this method is not performing a new inference on the text, but returns the predictions obtained during the training;
- `coherence` which computes the chosen coherence metric – among the ones described in Section 3.1 – on a given dataset;
- `evaluate` which evaluate the model predictions against a given ground truth, using the metrics described in Section 3.2.

[6] We use a Glove model pre-trained on Wikipedia 2014 + Gigaword 5, available at https://nlp.stanford.edu/projects/glove/

Algorithm	Acronym	Source implementation
Latent Dirichlet Allocation	LDA	http://mallet.cs.umass.edu/ (McCallum, 2002) (JAVA)
Latent Feature Topic Models	LFTM	https://github.com/datquocnguyen/LFTM (JAVA)
Doc2Topic	D2T	https://github.com/sronnqvist/doc2topic
Gibbs Sampling for a DMM	GSDMM	https://github.com/rwalk/gsdmm
Non-Negative Matrix Factorization	NMF	https://radimrehurek.com/gensim/models/nmf.html
Hierarchical Dirichlet Processing	HDP	https://radimrehurek.com/gensim/models/hdpmodel.html
Latent Semantic Indexing	LSI	https://radimrehurek.com/gensim/models/lsimodel.html
Paragraph Vector Topic Model	PVTM	https://github.com/davidlenz/pvtm
Context Topic Model	CTM	https://github.com/MilaNLProc/contextualized-topic-models

Table 1: Algorithms included in ToModAPI, with their source implementation. The original implementation of those model is in Python unless specified otherwise.

The structure of the library, which relies on class inheritance, is easy to extend with the addition of new models. In addition to allowing the import in any Python environment and use the library offline, it provides the possibility of automatically build a web API, in order to access to the different methods through HTTP calls. Table 2 provides a comparison between the ToModAPI, Gensim and STMM. Given that we wrap some Gensim models and methods (i.e. for coherence computation), some similarities between it and our work can be observed.

The software is distributed under an open source license[7]. A demo of the web API is available at http://hyperted.eurecom.fr/topic.

5 Datasets and pre-trained models

Together with the library, we provide pre-trained models trained on two different datasets having different characteristics (20NG and AFP). A common pre-processing is performed on the datasets before training, consisting of:

- Removing numbers, which, in general, do not contribute to the broad semantics;
- Removing the punctuation and lower-casing;
- Removing the standard English stop words;
- Lemmatisation using Wordnet, in order to deal with inflected forms as a single semantic item;
- Ignoring words with 2 letters or less. In facts, they are mainly residuals from removing punctuation – e.g. stripping punctuation from *people's* produces *people* and *s*.

The same pre-processing is also applied to the text before topic prediction.

5.1 20 NewsGroups

The 20 NewsGroups collection (20NG) (Lang, 1995) is a popular dataset used for text classification and clustering. It is composed of English news documents, distributed fairly equally across 20 different categories according to the subject of the text. We use a reduced version of this dataset[8], which excludes all the documents composed by the sole header while preserving an even partition over the 20 categories. This reduced dataset contains 11,314 documents. We pre-process the dataset in order to remove irrelevant metadata – consisting of email addresses and news feed identifiers – keeping just the textual content. The average number of words per document is 142.

5.2 Agence France Presse

The Agence France Presse (AFP) publishes daily up to 2000 news articles in 5 different languages[9], together with some metadata represented in the NewsML XML-based format. Each document is categorised using one or more subject codes, taken from the IPTC NewsCode Concept vocabulary[10]. In case of multiple subjects, they are ordered by relevance. In this work, we only consider the first level of the hierarchy of the IPTC subject codes. We extracted a dataset containing 125,516 news documents in English and corresponding to the production of AFP for the year 2019, with 237 words per document on average.

Table 3 summarizes the number of documents for each topic in those two datasets. In AFP, a single document can be assigned to multiple subject, so we take each assignment into account. The two

[7] https://github.com/D2KLab/ToModAPI

[8] https://github.com/selva86/datasets/

[9] The catalogue can be explored at http://medialab.afp.com/afp4w/

[10] http://cv.iptc.org/newscodes/subjectcode/

library	**Gensim**	**STMM**	**ToModAPI**
algorithms	8: LDA, LDA Sequence, LDA multicore, NMF, LSI, HDP, Author-topic model, DTM	11: LDA, LFTM, DMM, BTM, WNTM, PTM, SATM, ETM, GPU-DMM, GPU-PDMM, LF-DMM	9: LDA, LFTM, D2T, GSDMM, NMF, HDP, LSI, PVTM, CTM
language	Python	Java	Python
focus	general	short text	general
training	✓	✓	✓
inference	✓	✓	✓
corpus predictions	(by inferencing the corpus)	✓	✓
coherence metrics	c_{umass}, c_v, c_{uci}, c_{npmi}	c_{umass}	c_{umass}, c_v, c_{uci}, c_{npmi}
Evaluation with Ground Truth	-	purity, NMI	purity, homogeneity, completeness, v-measure, NMI
usage	import in script	command line	import in script, web API

Table 2: Comparison between topic modeling libraries. For details about the acronyms, refer to the documentation

datasets present multiple differences: total number of documents, distribution of documents per subject, and the fact that for AFP, one document can have multiple subjects.

20NG		AFP	
rec.sport.hockey	600	Politics	47277
soc.religion.christian	599	Sport	36901
rec.motorcycles	598	Economy, Business, Finance	31042
rec.sport.baseball	597	Unrest, Conflicts and War	21140
sci.crypt	595	Crime, Law and Justice	16977
sci.med	594	Art, Culture, Entertainment	8586
rec.autos	594	Social Issues	7609
comp.windows.x	593	Disasters and Accidents	5893
sci.space	593	Human Interest	4159
comp.os.ms-windows.misc	591	Environmental Issue	4036
sci.electronics	591	Science and Technology	3502
comp.sys.ibm.pc.hardware	590	Religion and Belief	3081
misc.forsale	585	Lifestyle and Leisure	3044
comp.graphics	584	Labour	2570
comp.sys.mac.hardware	578	Health	2535
talk.politics.mideast	564	Weather	1159
talk.politics.guns	546	Education	734
alt.atheism	480		
talk.politics.misc	465		
talk.religion.misc	377		
Total	11314	Total	125516

Table 3: Number of documents per subject in 20NG (20 topics) and AFP (17 topics)

5.3 Wikipedia Corpus

We also describe the Wikipedia corpus (Wiki)[11], which is a readily extracted and organised snapshot from 2013 that includes pages with at least 20 page views in English. This corpus has been used in other works, for example, for computing word embeddings (Leimeister and Wilson, 2018). The corpus is distributed with some pre-processing already applied, like lower-casing and punctuation stripping. However, we performed additional operations such as lemmatisation, stop-word and small word (2 characters or less) removal. The dataset consists of around 463k documents with 498M words. This corpus will not be used for training but only for evaluating the models (trained on 20NG or AFP) in order to reflect on the generalisation of the topics models.

6 Experiment and Results

We empirically evaluate the performances of the topic modeling algorithms described in Section 2 on the two datasets presented in Section 5 using the metrics detailed in Section 3. For each algorithm, we trained two different models, respectively on 20NG and AFP corpus. The number of topics – when required by the algorithm – has been set to 20 and 7 when training on 20NG and AFP, respectively, in order to mimic the original division in class labels of the corpora (except for GSDMM and HDP which infer the optimal number of topics). Each model trained on either 20NG or AFP is tested against the same dataset and the Wikipedia dataset to compute each metric.

Table 4 shows the average coherence scores of the topics computed on the 20NG dataset, together with the standard deviation, while the results of Table 5 refer to models computed on the AFP dataset. The results differ depending on the studied metric and the evaluation dataset. LFTM generalises better when evaluated against the Wikipedia corpus, probably thanks to the usage of pre-trained word vectors on large corpora. Overall, LDA has the best results on all metrics, always being among

[11] https://storage.googleapis.com/lateral-datadumps/wikipedia_utf8_filtered_20pageviews.csv.gz

	C_v				C_{NPMI}				C_{UMASS}				C_{UCI}			
	20NG		wiki		20NG		wiki		20NG		wiki		20NG		wiki	
CTM	0.56	(0.15)	0.46	(0.24)	-0.04	(0.19)	-0.06	(0.16)	-5.78	(5.27)	-4.28	(3.94)	-3.09	(4.18)	-2.51	(3.95)
D2T	0.57	(0.14)	0.51	(0.10)	0.01	(0.11)	0.05	(0.05)	-2.94	(1.67)	-2.02	(0.49)	-1.56	(2.39)	0.16	(0.81)
GSDMM	0.50	(0.18)	0.41	(0.20)	0.00	(0.19)	-0.04	(0.09)	-3.86	(2.88)	-2.45	(1.04)	-2.02	(3.16)	-1.44	(2.26)
HDP	0.44	(0.21)	0.48	(0.24)	-0.09	(0.17)	-0.04	(0.10)	-5.59	(5.04)	-3.25	(3.18)	-5.59	(5.04)	-2.21	(2.64)
LDA	**0.64**	(0.14)	0.55	(0.16)	**0.10**	(0.08)	**0.07**	(0.06)	-1.98	(0.68)	-1.75	(0.45)	**0.27**	(1.30)	0.53	(0.88)
LFTM	0.53	(0.09)	**0.56**	(0.17)	-0.01	(0.10)	**0.07**	(0.06)	-2.97	(3.15)	-1.72	(0.69)	-1.47	(2.47)	**0.58**	(0.76)
LSI	0.53	(0.22)	0.41	(0.11)	0.03	(0.16)	-0.04	(0.10)	-3.25	(2.16)	-2.64	(1.08)	-1.37	(2.89)	-1.69	(2.59)
NMF	0.61	(0.19)	0.52	(0.15)	0.10	(0.15)	-0.02	(0.12)	-2.37	(1.61)	-3.08	(4.83)	-0.03	(2.24)	-1.27	(2.97)
PVTM	0.54	(0.09)	0.46	(0.11)	0.06	(0.04)	0.04	(0.06)	**-1.63**	(0.82)	**-1.52**	(0.54)	0.21	(0.92)	0.25	(0.74)

Table 4: The mean and standard deviation of different coherence metrics computed on 2 reference corpora 20NG and Wikipedia. The models have been trained on 20NG.

	C_v				C_{NPMI}				C_{UMASS}				C_{UCI}			
	AFP		wiki		AFP		wiki		AFP		wiki		AFP		wiki	
CTM	0.54	(0.15)	0.56	(0.28)	-0.05	(0.17)	-0.04	(0.09)	-6.56	(5.94)	-3.47	(2.96)	-2.75	(3.73)	-1.49	(2.17)
D2T	0.58	(0.14)	0.45	(0.10)	0.06	(0.07)	-0.01	(0.07)	-2.25	(0.49)	-2.44	(0.73)	-0.02	(0.93)	-1.07	(1.42)
GSDMM	0.51	(0.12)	0.58	(0.17)	0.09	(0.07)	0.03	(0.11)	-1.72	(0.47)	-2.73	(1.31)	0.70	(0.66)	-0.29	(1.59)
HDP	0.42	(0.10)	**0.69**	(0.22)	0.02	(0.07)	0.01	(0.16)	-2.23	(0.92)	-2.74	(2.63)	-0.20	(1.05)	-0.63	(2.86)
LDA	0.65	(0.10)	0.54	(0.11)	0.11	(0.04)	**0.06**	(0.06)	-1.40	(0.23)	-1.88	(0.48)	0.80	(0.30)	**0.25**	(0.89)
LFTM	0.59	(0.14)	0.54	(0.20)	0.06	(0.10)	**0.06**	(0.12)	-1.97	(2.40)	-1.91	(2.19)	0.11	(2.08)	0.22	(2.58)
LSI	0.58	(0.12)	0.55	(0.14)	0.07	(0.09)	0.05	(0.11)	-1.80	(0.47)	-2.59	(1.37)	0.09	(0.96)	-0.36	(1.87)
NMF	**0.67**	(0.12)	0.46	(0.12)	**0.13**	(0.06)	0.04	(0.07)	-1.27	(0.29)	-1.73	(0.69)	**0.95**	(0.42)	0.07	(1.26)
PVTM	0.52	(0.12)	0.51	(0.09)	0.07	(0.06)	0.04	(0.04)	**-1.16**	(0.34)	**-1.56**	0.86	0.49	(0.41)	0.14	(0.63)

Table 5: The mean and standard deviation of different coherence metrics computed on 2 reference corpora AFP and Wikipedia. The models have been trained on AFP.

the top ones in terms of coherence. When trained on AFP, all topic models benefit of a bigger dataset; this results in generally higher scores and in different algorithms maximising specific metrics.

We also consider the time taken by the different techniques for different tasks like training and getting prediction (Table 6). The results have been collected selecting the best of 3 different calls. The inference time has been computed using the models trained on the 20NG dataset, on a small sentence of 18 words[12]. The table shows LDA leading in training, while the longest execution time belongs to LFTM. The inference time for all models is in the order of few seconds or even less than 1 for GSDMM, HDP, LSI and PVTM. The manipulation of BERT embeddings makes CTM inference more time-consuming. The inference timing for D2T is not computed because its implementation is not available yet.

7 Conclusions and Future Work

In this paper, we introduced ToModAPI, a library and a Web API to easily train, test and evaluate topic models. 9 algorithms are already included in the library, while new ones will be added in future. Other evaluation metrics for topic modeling have been proposed (Wallach et al., 2009) and will be included in the API for enabling a complete evaluation. Among these, metrics based on word embeddings are gaining particular attention (Ding et al., 2018). For further exploiting the advantage of having a common interface, we will study ways to automatically tune each model's hyper-parameters such as the right number of topics, find an appropriate label for the computed topics, optimise and use the models in real world applications. Finally, future work includes a deeper comparison of the models trained on different datasets.

	Training		Inference
	20NG	AFP	
CTM	544	9,262	19
D2T	192	5,892	-
GSDMM	1,194	21,881	0
HDP	430	7,020	0
LDA	80	1,334	2
LFTM	3,119	15,100	1
LSI	383	6,716	0
NMF	357	6,320	5
PVTM	193	3,757	0

Table 6: Model comparison from a time (in seconds) delay standpoint for training and inference.

[12] *"Climate change is a global environmental issue that is affecting the lands, the oceans, the animals, and humans"*

Acknowledgments

This work has been partially supported by the French National Research Agency (ANR) within the ASRAEL (grant number ANR-15-CE23-0018) and ANTRACT (grant number ANR-17-CE38-0010) projects, and by the European Union's Horizon 2020 research and innovation program within the MeMAD (grant agreement No. 780069) and SILKNOW (grant agreement No. 769504) projects.

References

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

David M. Blei and Jon D. McAuliffe. 2007. Supervised Topic Models. In 20th *International Conference on Neural Information Processing Systems (NIPS)*, pages 121—128.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993—-1022.

Ziqiang Cao, Sujian Li, Yang Liu, Wenjie Li, and Heng Ji. 2015. A Novel Neural Topic Model and Its Supervised Extension. In *AAAI Conference on Artificial Intelligence*.

Christian Chiarcos, Richard Eckart de Castilho, and Manfred Stede. 2009. *Von der Form zur Bedeutung: Texte automatisch verarbeiten - From Form to Meaning: Processing Texts Automatically*. Narr Francke Attempto Verlag GmbH + Co. KG.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Ran Ding, Ramesh Nallapati, and Bing Xiang. 2018. Coherence-Aware Neural Topic Modeling. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 830–836, Brussels, Belgium.

Anjie Fang, Craig Macdonald, Iadh Ounis, and Philip Habel. 2016. Using Word Embedding to Evaluate the Coherence of Topics from Twitter Data. In *39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1057—1060.

Thomas S. Ferguson. 1973. A bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1(2):209–230.

Malek Hajjem and Chiraz Latiri. 2017. Combining IR and LDA Topic Modeling for Filtering Microblogs. In *21st International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES)*, pages 761–770, Marseille, France.

Christoph Kling. 2016. *Probabilistic models for context in social media*. doctoral thesis, Universität Koblenz-Landau, Universitätsbibliothek.

Andrea Lancichinetti, Santo Fortunato, and János Kertész. 2009. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3).

Ken Lang. 1995. NewsWeeder: Learning to Filter Netnews. In *20th International Conference on Machine Learning (ICML)*, pages 331–339.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *31st International Conference on Machine Learning (ICML)*, pages 1188–1196, Bejing, China.

Matthias Leimeister and Benjamin J. Wilson. 2018. Skip-gram word embeddings in hyperbolic space. Arxiv.

David Lenz and Peter Winker. 2020. Measuring the diffusion of innovations with paragraph vector topic models. *PLOS ONE*, 15:1–18.

Tapio Luostarinen and Oskar Kohonen. 2013. Using Topic Models in Content-Based News Recommender Systems. In *19th Nordic Conference of Computational Linguistics (NODALIDA)*.

Andrew Kachites McCallum. 2002. MALLET: A Machine Learning for Language Toolkit.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *26th International Conference on Neural Information Processing Systems (NIPS)*, volume 2, pages 3111–3119, Lake Tahoe, NV, USA.

David Newman, Chaitanya Chemudugunta, Padhraic Smyth, and Mark Steyvers. 2006. Analyzing Entities and Topics in News Articles Using Statistical Topic Models. In *Intelligence and Security Informatics*, pages 93–104.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving Topic Models with Latent Feature Word Representations. *Transactions of the Association for Computational Linguistics*, 3:299–313.

Derek O'Callaghan, Derek Greene, Joe Carthy, and Pádraig Cunningham. 2015. An analysis of the coherence of descriptors in topic modeling. *Expert Systems with Applications*, 42(13):5645–5657.

Pentti Paatero and Unto Tapper. 1994. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Jipeng Qiang, Zhenyu Qian, Yun Li, Yunhao Yuan, and Xindong Wu. 2019. Short Text Topic Modeling Techniques, Applications, and Performance: A Survey. Arxiv.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *LREC Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3982–3992, Hong Kong, China.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *8th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 399––408.

Andrew Rosenberg and Julia Hirschberg. 2007. V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic.

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

Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. 2009. Evaluation methods for topic models. In *26th Annual International Conference on Machine Learning (ICML)*, pages 1105––1112.

Jianhua Yin and Jianyong Wang. 2014. A Dirichlet Multinomial Mixture Model-Based Approach for Short Text Clustering. In *20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 233––242.