Useful ToPIC: Self-tuning strategies to enhance Latent Dirichlet Allocation

Useful ToPIC: Self-tuning strategies to enhance Latent Dirichlet Allocation

Stefano Proto, Evelina Di Corso, Francesco Ventura and Tania Cerquitelli
Dipartimento di Automatica e Informatica
Politecnico di Torino
Torino, Italy
Email: name.surname@polito.it

***Abstract*—ToPIC (Tuning of Parameters for Inference of Concepts) is a distributed self-tuning engine whose aim is to cluster collections of textual data into correlated groups of documents through a topic modeling methodology (i.e., LDA). ToPIC includes automatic strategies to relieve the end-user of the burden of selecting proper values for the overall analytics process. ToPIC's current implementation runs on Apache Spark, a state-of-the-art distributed computing framework. As a case study, ToPIC has been validated on three real collections of textual documents characterized by different distributions. The experimental results show the effectiveness and efficiency of the proposed solution in analyzing collections of documents without tuning algorithm parameters and in discovering cohesive and well-separated groups of documents with a similar topic.**

***Keywords*-Text mining; Parameter-free technique; topic detection; LDA; data weighting function; Big Data framework.**

I. Introduction

With modern applications and technologies (e.g., social networks, e-learning platforms, digital libraries), the production and collection of textual data is constantly increasing. However, the sheer volume of documents means that extracting useful information from the data can be difficult and challenging. It is therefore increasingly necessary to engineer effective approaches to automatically discover hidden and implicit information from textual data. Text mining, and specifically topic modeling, studies algorithms to find previously unknown but potentially high-quality information from large document collections. Text mining activities include: (i) grouping documents with similar properties or similar content [1], (ii) topic modeling [2], (iii) Clustering Web services [3] and (iv) long text summarizations [4].

Topic models are a set of statistical text-mining tools that uncover the hidden topics in a collection of documents. Intuitively, documents can be associated with a particular topic or can be seen as a mixture of topics in different proportions, and certain words can be expected to appear in a document more or less frequently. Topic modeling techniques are then able to describe topics (and so documents) by means of similar word clusters.

Analyzing large data collections with the above techniques often entails a critical bottleneck of computational costs. Several studies have sought solutions to this and their results have produced innovative algorithms and methodologies able to support large scale analytics, such as MapReduce [5] and particularly Apache Spark [6], which outperforms others (e.g., Hadoop) thanks to its distributed memory abstraction. Applying these techniques to text mining becomes natural in view of the large volume of the data collections. Besides the high computational costs, effectively performing the text mining process on textual data currently requires a multi-step process involving various algorithms with each one having different specific parameters that should be manually set by the analyst. This activity is very time consuming and requires a lot of expertise to achieve the best trade-off between the quality of the result and execution time. Innovative, scalable, and parameter-free solutions need to be devised to streamline the analytics process for large data collections.

This paper proposes ToPIC (Tuning Of Parameters for Inference of Concepts), a distributed self-tuning engine running on Apache Spark, which is able to cluster a collection of documents into cohesive and well-separated groups. It includes all the analytics blocks to make the overall analysis problem more effectively tractable. Specifically, ToPIC comprises different suitable data weighting functions, based on local and global weights, together with the application of the Latent Dirichlet Allocation [2], a generative probabilistic model described in Section III, to divide a given corpus into correlated groups of documents with a similar topic. Moreover, ToPIC contains a procedure, named ToPIC-Similarity, to relieve the end-user of the burden of selecting proper values for the number of topics. We have experimentally assessed ToPIC on three real collections of documents: two Wikipedia textual data collections and the Reuters-21578 collection. The performed experiments highlighted ToPIC's ability to autonomously identify groups of documents with a similar topic by off-loading the parameter tuning from the end user.

In this paper Section II explores the state-of-the-art techniques, Section III presents ToPIC architecture and its main building components, while Section IV shows the tests run to assess ToPIC's performance and discusses their experimental results. Finally, Section V draws conclusions and presents future developments of this work.

II. Related works

Finding and setting the optimal number of topics in which to cluster the documents is an open issue in literature. Besides the trial-and-error approach, which requires a lot of effort

and expertise to correctly configure each algorithm, different methodologies have been proposed. Among these, the approaches considered in this study are (i) *En-LDA* and (ii) *RPC*.

The *Entropy optimized Latent Dirichlet Allocation* (En-LDA) approach [7] is an entropy based index that chooses the K number of clusters to reduce the model entropy as much as possible. This method measures the entropy of the LDA model considering the entropy of each term in a document and using topics as their probabilistic labels. The entropy of words in all the corpus under analysis are then aggregated to estimate the overall entropy of words, given the distribution of words with respect to topics and the distribution of topics with respect to documents.

The *Rate of Perplexity Change* (RPC) [8] uses an heuristic approach that takes into account how the variation of the average perplexities for a set of topic candidates changes with respect to the number of topics. The perplexity index [2] measures how well a probability model predicts a sample. Thus, the RPC index is defined as the absolute value of the ratio between the difference of perplexity values for different models and the difference between the number of topics in the corresponding models. Authors in [8] consider the first change point of the RPC curve, which is the first value in the range $2 \leq i \leq K-1$ that satisfies the equation $RPC(i) < RPC(i+1)$, to be the most suitable value for the number of topics K.

With respect to these approaches that consider only the probabilistic properties of the LDA models, ToPIC includes in the computation also the semantic representation of the identified clusters. Moreover, it appears that the variances of both perplexity and entropy are very low, considering different K values for the same dataset. Thus, choosing a K value rather than another based only on probability statistics may be not effective for the analysis. Furthermore, the evaluation of the En-LDA does not have a stopping criterion to interrupt the computation to find the optimal K.

Self-tuning strategies to automatically tune the algorithm parameters for the whole text clustering process have been proposed in [1], [9]. In [1], the authors explore a data transformation method of input data (Latent Semantic Indexing) before the cluster analysis (addressed via the K-Means algorithm) to gain insights from textual data and make the overall analysis problem more effectively tractable. The contribution proposed in [9] tailors the methodology in [1] to Twitter data.

Different from the above works, this study proposes a new method to automatically set the optimal number of topics to partition a given document collection through LDA, exploiting several weighting schemas and datasets characterized by different data distribution.

III. Methodology

ToPIC (Tuning Of Parameters for Inference of Concepts) is a distributed self-tuning engine whose aim is to cluster collections of textual documents into correlated groups of documents through a topic modeling methodology. ToPIC includes automatic strategies to relieve the end-user of the burden of selecting proper values for the overall cluster analysis process. The ToPIC architecture, reported in Figure 1, includes three main components: (i) *Textual data processing and features computation*, (ii) *Document modeling and self-tuning textual data clustering* and (iii) *Knowledge visualization and validation*.

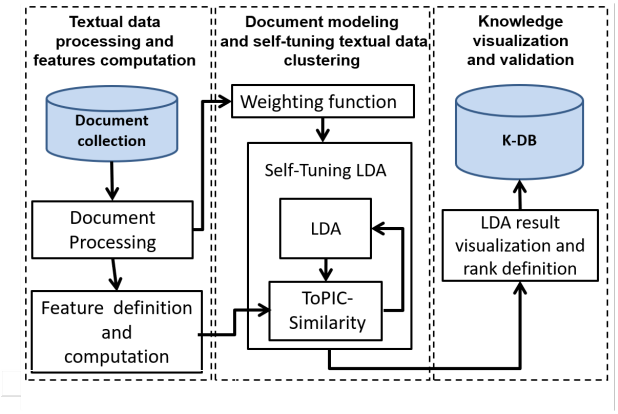

Fig. 1. ToPIC architecture

Textual data processing is carried out through five steps which are done sequentially as interrelated tasks. (1) *Document splitting*: documents can be split into sentences, paragraphs, or analyzed in their entire content, according to the next analytics task; (2) *tokenization*: each document is broken into groups of words named tokens within the same split; (3) *stopwords removal*: the most common words in a language (e.g., articles, prepositions), which do not bring additional information, are eliminated; (4) *stemming*: prefixes and suffixes are removed to reduce each word to its base or root form; (5) *case normalization*: this last step converts the text to uppercase or lowercase.

Features definition and computation is an engine able to characterize the data distribution of a collection of documents through several indices as proposed in [10], [11]. It includes: number of categories; number of documents, maximum, minimum and average frequency of a term's occurrence; number of terms in the collection with repetitions (# term), number of different terms in the collection without repetition (dictionary); the ratio between the dictionary variety and the total number of terms in the collection (Type-Token Ratio); the ratio between number of Hapax (absolute frequency of terms with one occurrence) and the cardinality of the dictionary (Hapax %); and the ratio between the cardinality of the dictionary and the square root of # terms (Guiraud Index).

A. Document modeling and Self-tuning textual data clustering

This ToPIC component entails the representation of the document collection through different weighting functions to highlight the relevance of specific terms in the collection, and then it applies LDA to build a model able to cluster a given collection of textual data into correlated groups of documents with a similar topic.

TABLE I
LOCAL AND GLOBAL WEIGHT FUNCTIONS EXPLOITED IN TOPIC

Weight	Definition
Local	TF = tf_{ij}
	LogTF = $\log_2(\mathrm{tf}_{ij}+1)$
	Boolean = $\{0, 1\}$
Global	IDF = $\log \frac{\lvert D\rvert}{\mathrm{df}_j}$
	Entropy = $1+\sum_i \frac{p_{ij}\log p_{ij}}{\log n}$
	TF_{glob} = tf_j

Let $\mathcal{D}$ = $\{\mathbf{w}_1, \mathbf{w}_2, \ldots \mathbf{w}_M\}$ be a collection of *M* documents, named *corpus*, $\mathbf{w}_i$ = (w_1, w_2, ... w_N) a general document in it and $V = \{t_1, t_2, \ldots, t_{|V|}\}$ the set of distinct terms in the collection, i.e. the set of all the distinct stems used at least once in a document. $\mathcal{D}$ is represented as a matrix X, named document-term matrix. In X, rows correspond to documents in the corpus and columns, one for each $t_j \in V$, correspond to terms.

Weighting functions. To measure the relevance of terms/ words appearing in the document, each term in the corpus $\mathcal{D}$ is associated with a *weight*. A weight w_{ij} is a positive real number associated with each term t_j of a document d_i, and quantifies its level of importance in each document under analysis. Various weighting functions, combining a local term weight with a global term weight, have been proposed in [12]. A weighting function applied on a collection $\mathcal{D}$ generates its weighted matrix X. Specifically, for each term t_j of a document d_i the corresponding weight x_{ij} in X is computed as the product of a local term weight (l_{ij}) and a global term weight (g_j) (x_{ij}=$l_{ij} \times g_j$). l_{ij} measures the relative frequency of a term j in a document i, while g_j describes the relative frequency of the term t_j within the entire collection D. TOPIC includes three local term weights: *Term-Frequency* (TF), *Logarithmic term frequency* (Log) and *Unitary* (Boolean), and three global term weights: *Inverse Document Frequency* (IDF), *Entropy* (Entropy) and *Term-Frequency* (TF_{glob}). The formulas for all local and global weights exploited in TOPIC are shown in Table I. TOPIC investigates the five weighting schema as TF-IDF, TF-Entropy, Log-IDF, Log-Entropy, Boolean-TF_{glob}.

B. Self-Tuning LDA

This TOPIC component discovers a good partition of a document collection based on the LDA modeling. The latter produces a probabilistic model evaluated by the TOPIC-SIMILARITY block. The result of TOPIC-SIMILARITY is then fed back into the LDA algorithm to produce a new evaluation, taking into account a new configuration and the previous TOPIC-SIMILARITY evaluation. This process is repeated until a good trade-off between the quality of the obtained results and the execution time is achieved.

Latent Dirichlet Allocation (LDA) is an unsupervised generative probabilistic model for collections of discrete data such as text corpora [2]. The aim of this algorithm is to build a model able to shortly describe large collections of data without resorting to data dimensionality reduction. Given the document-term matrix of weights X the generative model is computed for a specific number of topics *K*, which is supposed to be known. However, in many real use cases the right value for *K* is unknown and it is set experimentally. TOPIC allows the automatic identifying of a good value for *K* through TOPIC-SIMILARITY. The LDA modeling describes topics and words as probabilistic distributions from which the document terms will be drawn. Documents are then seen as a distribution over a mixture of latent topics, since each term of a document is drawn from the vocabulary taking into account the terms' probabilities for each given topic of the document's mixture [2]. Specifically, to generate each document in the corpus, the steps performed are:

1) Choose the number of terms from a Poisson distribution;
2) For each of the document's words:
   - Choose a topic z_n from Multinomial($\boldsymbol{\theta}$), where $\boldsymbol{\theta}$ is a Dirichlet($\boldsymbol{\alpha}$), representing the document-topics distribution;
   - Choose a word w_n from Multinomial($\boldsymbol{\phi}_{z_n}$), where $\boldsymbol{\phi}$ represents the topic-words distribution ($\boldsymbol{\phi} \sim$ Dirichlet($\boldsymbol{\beta}$)), conditioned on the topic z_n.

Hence, the joint multivariate distribution for the whole corpus of the document-topics distribution $\boldsymbol{\theta}$, the set of *K* topics $\mathbf{z}$ and the set of *N* terms $\mathbf{w}$ is defined as: $p(\mathcal{D}|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod_{d=1}^{K} \int p(\boldsymbol{\theta}_d|\boldsymbol{\alpha}) \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_n|\boldsymbol{\theta}_d) p(w_{dn}|z_{dn}, \boldsymbol{\beta})\right) d\boldsymbol{\theta}_{\mathrm{d}}$, where

- $\boldsymbol{\alpha}$ represents the concentration for the prior placed on documents' distributions over topics;
- $\boldsymbol{\beta}$ describes the concentration for the prior placed on topics' distributions over terms.

With a corpus of documents X in input, the generative LDA model can then be used to support inference of the posterior distribution of the latent variables for the given corpus. Generally, computing these distributions it is unfeasible, and so it is impossible to exactly solve this posterior Bayesian inferential problem. To overcome this problem, several approximate inference algorithms have been proposed in literature: the TOPIC engine exploits the Online Variational Bayes algorithms [13], while $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are set to maximize the log likelihood of the data under analysis.

ToPIC-SIMILARITY index. In order to predict a suitable number of topics (i.e. the desired number of clusters) to divide the corpus into, TOPIC uses a novel proposed strategy named TOPIC-SIMILARITY to assess how topics are semantically diverse and choose proper configurations for the LDA modeling. Given a lower and an upper bound number of clusters set by the analyst (i.e, [K_{min}, K_{max}]) a new LDA model is generated for each *K* value. For each of these partitionings, TOPIC-SIMILARITY requires three steps to be gone through:

1) *topic characterization*, to describe each *t* topic ($0 \leq t <$ *K*) with the most *n* representative words;
2) *similarity computation*, to assess how the topics in the same partitioning are similar;
3) *K identification*, to find good clustering configurations to be proposed to the analyst;

Steps 1) and 2) are repeated for all the *t* topics in every *K* clustering model.

Topic characterization. To determine the TOPIC-SIMILARITY, each topic *t* has to be described with its *n* most representative terms. This number of words is automatically set, depending on:

- variety of the corpus dictionary $|V|$,
- the average frequency of the terms in the corpus (*AvgFreq*),
- the Type-Token Ratio (*TTR*, $\in$ [0, 1]) and the currently considered number of topics *K*.

To select the number of representative words, TOPIC considers only the richest part of the corpus under analysis (by means of the TTR index, which represents the lexical variation of the corpus) and then samples the remaining words by the average frequency of the terms. This quantity, named Q, represents the total number of considered terms, and it is mathematically defined as: $Q := \frac{|V| \cdot TTR}{AvgFreq}$. Given Q, the number of words *n* describing a single topic *t* is given by:

$$n = \begin{cases} \frac{Q}{K}\ , & if\ Q \geq K \cdot AvgFreq \\ AvgFreq\ , & if\ Q < K \cdot AvgFreq \end{cases} \quad (1)$$

For each topic *t*, the number of *n* terms is obtained taking the corpus dictionary sorted by the distribution ϕ.
This is done if the final number of words considered for each topic is greater than the average frequency of terms of the corpus, to make the sampling reasonable (it does not make sense to sample with a period greater than the number of items themselves) and to have every topic represented at least by a number of words equal to the average frequency. If the condition is not satisfied, then a minimum number of terms is considered. We set this lower bound quantity to be equal to the average frequency of the terms in the corpus. Repetitions among the considered terms characterizing each topic are removed and the resulting words are considered together to make the topic representations comparable. Then, for each term in every topic, if the word is present in the topic description, the correspondent value is set to the probability that the term has to be picked up in the topic, or to 0 if it is not.

Similarity computation To compute the TOPIC-SIMILARITY index of the partitioning, all the possible pairs of topics are considered. Similarity among all the topics is computed through the cosine similarity. Cosine similarity is one of the most used techniques in information retrieval and data mining, especially in text analysis and topic modeling, because of its efficiency [14] and ability to reflect the human perception of similarity [15]. It is derived from the Euclidean dot product and, given two topics *t'* and *t"* of the same partitioning *K*, it is computed as follows: $similarity(t', t'') = \frac{\mathbf{N}_{t'} \cdot \mathbf{N}_{t''}}{\|\mathbf{N}_{t'}\|_2 \|\mathbf{N}_{t''}\|_2}$, where $\mathbf{N}_{t'}$ and $\mathbf{N}_{t''}$ are the set of representative words of topic *t'* and *t"*, respectively. The result is a $K \times K$ symmetric matrix where each cell (*i,j*) is the similarity between topic in row i and topic in column j. Since we use the cosine similarity in the text analysis context and the probabilities of the terms are always positive, the obtained values will always be in the interval from 0 to 1. The TOPIC-SIMILARITY index for the considered clustering is obtained averaging the similarity matrix over *K* to keep in consideration the different number of clusters. For this issue, the norm of the whole similarity matrix (using the Frobenius norm) is computed and then the obtained values are divided by *K*. Since TOPIC-SIMILARITY is expressed in percentage, the previous values are multiplied by 100.

K identification A topic similarity function is obtained, computing TOPIC-SIMILARITY for the several LDA models generated for different *K*. To find optimal *K* values a trade-off approach between optimal results and computational cost has been chosen. It has been empirically seen that the obtained TOPIC-SIMILARITY function is, in most cases, decreasing but not monotonic. Two conditions are defined to choose as *K* values:

- the local minima of the curve, namely the *K* for which TOPIC-SIMILARITY(K_i) $<$ TOPIC-SIMILARITY(K_{i+1})
- the only points belonging to a decreasing segment of the curve. Thus, the second derivative is computed and only the points that have a positive second derivative are considered.

In our study we considered the selected values to be the first three points that satisfy both of the above conditions. The topic modeling and the search for optimal *K* values can stop when the first three values are found, or when the algorithm reaches the *K* upper bound value set by the analyst (and in this case a lower number of optimal values will be proposed to the analyst).

C. Knowledge visualization and validation

Model evaluation is hard when using unlabeled data. Since it is important to check whether a model makes sense practically, four different methods are integrated in TOPIC to assess the model quality. TOPIC includes two *visualization approaches*: (i) t-SNE visualization and (ii) topic-term diagram and two *quantitative validation* approaches: (iii) perplexity and (iv) clustering metrics. All the approaches are described below.

t-SNE. Visualizing high-dimensional data, such as textual documents, is not a trivial task. TOPIC includes *t-Distributed Stochastic Neighbor Embedding* (t-SNE) [16], a dimensionality reduction algorithm for high-dimensional data visualization. It allows the representing of data in two or three dimensions while preserving their most significant structure. Although t-SNE modeling is nondeterministic and probabilistic, it provides a good approximation of the underlying data.

Topic-Term representation. Since the topics descriptions are available after the LDA modeling process, TOPIC allows the comparison across and within the topics identified by the model through the analysis of the most representative terms and their probabilities. TOPIC provides the word cloud representation to visualize the topic-terms distribution. Besides showing the content of a topic, this visualization approach

stresses the terms with higher probabilities to represent the topic with a bigger font size. Thus, it is possible to straightforwardly observe whether the identified partition is good or the topic modeling has not yielded an acceptable outcome.

Perplexity. TOPIC includes the perplexity [2] measure to assess the quality of the probabilistic model. It describes how well a model depicts a sample, i.e. how much it is perplexed by a sample from the observed data. The lower the perplexity score, the better the model for the given data.

Clustering quality metrics. To evaluate the goodness of the LDA model, TOPIC integrates two metrics: (i) *Entropy* and (ii) *Silhouette*. In information theory, Entropy [17] is defined as the amount of information in a transmitted message. The larger the entropy, the more uncertainty is contained in the message. Instead, to measure the consistency and the cohesion of a clustering, Silhouette [18] is one of the most used and well-known metrics able to assess cluster separation together with cluster cohesion. It can assume a value between -1 and 1, representing how well the clustering performs for the given dataset.

IV. EXPERIMENTAL RESULTS

Experimental validation has been designed to address two main issues: (i) TOPIC's performance and (ii) comparison with state-of-the-art approaches. The current implementation of TOPIC is a project developed in Scala exploiting the LDA algorithm available in apache.spark.ml.clustering [19]. The two hyper parameters α and β have been set to $50/k$ and to 0.1 respectively, as proposed in [20]. All experiments have been performed on the BigData@PoliTO cluster[1] running Apache Spark 2.0.0. Three nodes, two executors and a driver having a 7GB main memory and a quadcore processor, have been deployed for this research using a cluster configuration.

A. Datasets

We experimentally assessed TOPIC on three real collections of documents: two Wikipedia textual data collections (including respectively 5 categories[2] and 10 categories[3]) and the Reuters-21578 collection[4]. The main characteristics of each dataset are reported in Table II.

Table II shows the proposed indices characterizing the data distribution in the considered datasets with Hapax (WH) and without Hapax (WoH). Eliminating words that appear only once within the corpus (i.e., Hapax) allows the LDA to construct a more precise probabilistic model. Since each term of the corpus is drawn from the vocabulary taking into account the terms' probabilities for each given topic of the documents' mixture, excluding Hapax terms will allow the reduction of noise in the vocabulary. Moreover, the features with Hapax (WH) and without Hapax (WoH) in Table II does not show any significant difference. Thus, removing Hapax does not change the main dataset features but allows better LDA modeling.

The TTR index well describes the data sparsity of each dataset and it is able to distinguish between sparse and dense data distribution. In these datasets, the TTR index falls into the range [0.01, 0.03]. Thus, dataset D3 is denser than the other two. Conversely, the Guiraud index is able to describe the lexical richness of a document corpus. D1 and D2 are characterized by a large number of terms (words with repetition) and by a very large dictionary (number of distinct words without repetition). Each word appears on average 16 times and 21 times respectively (see columns WH in Table II), but over 50% of words in both corpus appear only once (percentage of Hapax). Removing Hapax from the analysis increases the value of the average frequency in D1 and D2 to 32 and 44, respectively. On the other hand, D3 is characterized by a smaller dictionary, and the Hapax percentage is smaller than D1 and D2. Thus, the lexical richness used in D3 is not significant with respect to the other two datasets. These proportions are also confirmed excluding Hapax terms.

B. ToPIC's performance

Here we discuss TOPIC's ability to discover good quality partitions of a collection of textual documents. Table III shows the results obtained using TOPIC for each dataset and weighting schema. Since weights highlight the importance of words within documents, analyzing how different weighting schemas affect the model is important. Specifically, Table III includes a row for each K obtained utilizing our proposed clustering methodology and the three proposed metrics used to analyze the goodness of the statistical model generated. The selection of the three K values is performed through the analysis of the TOPIC-SIMILARITY curve.

An analysis of TOPIC's performance, reported in Table III, reveals several trends. Entropy and Silhouette are inversely proportional, since better clustering is characterized by a high level of Silhouette and a small value of entropy. Specifically, local weight TF tends to find a smaller number of topics using TOPIC-SIMILARITY independently of the global weight. Instead, local weight LogTF finds a large number of clusters which makes it possible to analyze the same dataset in a more detailed way that is able to find some interesting subtopics. Different considerations arise for the global weights. Global IDF results present a better value of perplexity (e.g., at least 0.1 greater) compared to those obtained using global Entropy, although the quality metrics are not in line. However, analyzing the found topics only by means of the quantitative metrics is not enough, since they merely measure distances (Euclidean and probabilistic) among the document groups. A deep understanding of human common-sense knowledge to interpret the major topic in each cluster should be done in order to validate the LDA model. Moreover, since TOPIC-SIMILARITY proposes three good values for the analysis, the analyst can choose a solution from the results according to the required granularity of the topics.

[1] https://bigdata.polito.it/content/bigdata-cluster

[2] Cooking, literature, mathematics, music and sports.

[3] Astronomy, cooking, geography, history, literature, mathematics, music, politics, religion, sports.

[4] http://www.daviddlewis.com/resources/testcollections/reuters21578. A subset of the collection has been taken, namely the *Apte' Split 90 categories* (http://disi.unitn.it/moschitti/corpora.htm)

TABLE II
DATASETS FEATURES WITH HAPAX (WH) AND WITHOUT HAPAX (WOH)

	Wikipedia				Reuters	
	WH	**WoH**	**WH**	**WoH**	**WH**	**WoH**
ID	D1		D2		D3	
# categories	5		10		90	
# documents	989		2,463		15,437	
Max frequency	6,343		9,405		42,886	
Min frequency	1	2	1	2	1	2
Avg frequency	16	32	21	44	55	76
# terms	1,109,408	1,075,210	2,995,762	2,882,781	1,337,225	1,316,988
Dictionary $\|V\|$	67,613	33,415	138,329	65,336	24,239	17,153
TTR	0.06	0.03	0.05	0.02	0.02	0.01
Hapax %	50.58	0	52.77	0	29.2	0
Guiraud Index	64.19	32.23	80.46	38.48	20.96	14,95

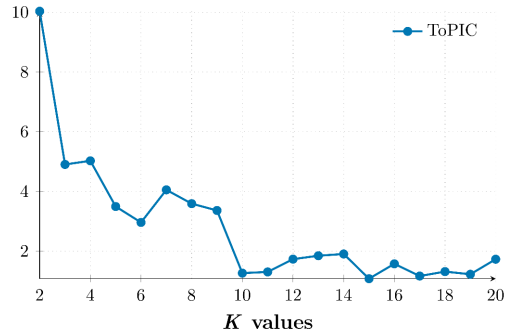

Fig. 2. TOPIC-SIMILARITY curve for dataset D1

Figure 2 shows the TOPIC-SIMILARITY curve for dataset D1 and TF-IDF weighting schema. The first three points that satisfy both the conditions stated in Section III-B are 3, 6 and 10. These three values are selected by TOPIC and reported in Table III.

D1 is discussed as the representative dataset. We computed the histogram of the TF-IDF and LogTF-Entropy weights. The values of LogTF-Entropy present almost a uniform distribution in the range [0,1] (Kurtosis index $>$ 0 and standard dev. 0.5) and the distribution has maximum value 8. Instead, with the IDF an asymmetrical bell distribution is obtained with average values between [2,5] (Kurtosis index $>$ 0 and standard dev. 12.7) and maximum value 1161. The IDF weight schema better differentiates the weights within the corpus, thus producing a probabilistic model with better performances. The Entropy global weight performs badly in providing relevance for the words in all the datasets. As shown in Figure 3, even though the quantitative evaluation metrics cannot spot the bad results of the clustering produced with these weighting schemas, it is still possible to asses the quality of the generated model. Indeed, Figure 3 presents interesting considerations analyzing dataset D1 for LogTF-Entropy (on top) and TF-IDF (on bottom) schemas. Specifically, figure 3(left) shows for the LDA models (obtained with the two weights) the probability distribution of each document of the corpus D1 to belong to the K topics selected by the algorithm (i.e., K=6 for TF-IDF and K=7 for LogTF-Entropy). In detail, the documents present a more homogeneous distribution using the IDF weight, with topics balanced by the number of documents. Instead, with the Entropy weight, there is one cluster in which 90% of the documents have a probability greater than 0.90 of membership. It turns out that 90% of the documents belong to a single cluster (topic) and the result is due to the fact that the weight entropy fails to isolate the most significant terms within the collection of documents. These results are also confirmed analyzing the t-SNE visualization in Figure 3(right). For this reason, clustering results having Entropy as global weights will not be considered in the discussion. When unbalanced clusters are generated, the use of only goodness metrics is not able to guarantee good performance. Indeed, high values of Silhouette or low values of entropy do not involve a good clustering but represent a simplification of the problem. It is like classifying 90% of the documents in a single topic, thus generating many false negatives. Having the class label available, indices such as recall or precision could help identify these incorrect assignments. However, if the label were not available, the use of quantitative indicators would not be effective. Methods that take into account the semantics must be presented.

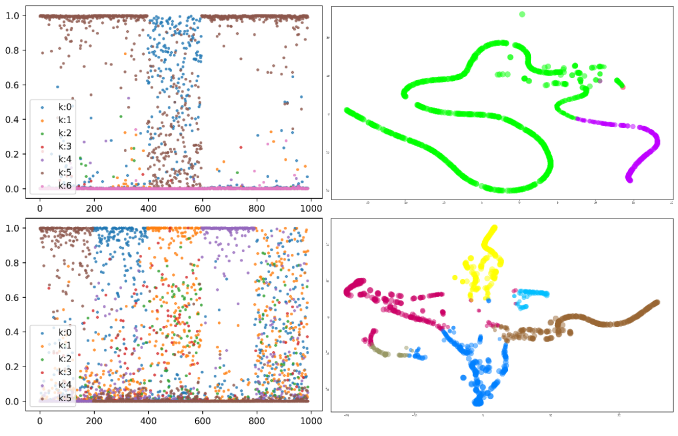

Fig. 3. (left) Document probability distributions in each topic (right) t-SNE representations, for dataset D1

The TOPIC performances are reported in the last column of Table III. The execution times for the used datasets are

TABLE III
TOPIC'S PERFORMANCE

	Weight	K	Perpl	Silh	Entr	Execution Time
D1	TF-IDF	3	8.8127	0.7721	0.2561	1h 50min
		6	8.5970	0.6935	0.3634	
		10	**8.4822**	**0.6827**	**0.3956**	
	TF-Entr	**5**	**9.0724**	**0.7623**	**0.2825**	1h 30min
		8	9.2482	0.6324	0.3388	
		9	9.2679	0.6319	0.3395	
	LogTF-IDF	**8**	**9.1873**	**0.6754**	**0.3205**	1h 40min
		17	9.1262	0.6370	0.3626	
	LogTF-Entr	5	9.9126	0.8915	0.1004	1h 30min
		7	**9.8841**	**0.8460**	**0.1748**	
		11	9.9794	0.9515	0.1089	
	Boolean-TF	4	6.4926	0.6979	0.4214	2h 5min
		5	**6.4640**	**0.6618**	**0.4832**	
		17	6.4208	0.3813	1.0901	
D2	TF-IDF	3	9.2008	0.7715	0.2460	3h 20min
		8	8.9628	0.5878	0.5314	
		10	**8.9436**	**0.5530**	**0.6118**	
	TF-Entr	3	9.5568	0.8075	0.2161	3h 25min
		7	**9.4555**	**0.7008**	**0.3556**	
		8	9.4631	0.6985	0.3693	
	LogTF-IDF	**11**	**9.4108**	**0.6016**	**0.4895**	3h 20min
		14	9.4529	0.5652	0.4958	
	LogTF-Entr	**7**	**10.2031**	**0.8751**	**0.1258**	3h 10min
		9	10.2194	0.8922	0.1219	
		11	10.2327	0.9012	0.1253	
	Boolean-TF	6	6.6223	0.4398	0.7979	5h 20min
		13	6.5833	0.3380	1.1922	
		18	**6.5699**	**0.3205**	**1.3262**	
D3	TF-IDF	3	7.7154	0.7347	0.2763	1h 55min
		4	7.6455	0.6913	0.3485	
		9	**7.4389**	**0.5966**	**0.5586**	
	TF-Entr	4	8.5396	0.0564	1.3806	1h 50min
		6	8.6242	-0.247	1.7805	
		9	**8.7109**	**-0.0811**	**2.169**	
	LogTF-IDF	5	7.7503	0.7005	0.3565	1h 50min
		7	7.6686	0.6676	0.4440	
		13	**7.5614**	**0.5989**	**0.6396**	
	LogTF-Entr	**5**	**8.7880**	**0.0774**	**1.6090**	1h 50min
		9	9.0011	-0.0437	2.1955	
		13	9.1759	-0.0690	2.5600	
	Boolean-TF	4	3.9669	0.6373	0.4659	2h 20min
		7	3.8947	0.4730	0.7352	
		16	**3.7309**	**0.3016**	**1.3112**	

TABLE IV
PERFORMANCE OF STATE-OF-THE-ART METHODS VS TOPIC

	Weights	Method	K	Perpl	Silh	Entr
D1	TF-IDF	RPC	3	8.8127	0.7721	0.2561
		En-LDA	19	8.4273	0.6211	0.5345
		TOPIC	10	8.4822	0.6827	0.3956
	TF-Entr	RPC	5	9.0724	0.7623	0.2825
		En-LDA	5	9.0724	0.7623	0.2825
		TOPIC	5	9.0724	0.7623	0.2825
	LogTF-IDF	RPC	7	9.1837	0.6935	0.3192
		En-LDA	16	9.1890	0.5530	0.4434
		TOPIC	8	9.1873	0.6754	0.3205
	LogTF-Entr	RPC	3	9.7774	0.8524	0.1441
		En-LDA	3	9.7774	0.8524	0.1441
		TOPIC	7	9.8841	0.8460	0.1748
	Boolean-TF	RPC	4	6.4926	0.6979	0.4214
		En-LDA	20	6.4127	0.6618	1.2558
		TOPIC	5	6.4640	0.6618	0.4832
D2	TF-IDF	RPC	6	9.0370	0.6214	0.4638
		En-LDA	20	8.8447	0.5274	0.6617
		TOPIC	10	8.9436	0.5530	0.6118
	TF-Entr	RPC	5	9.4955	0.7198	0.3293
		En-LDA	20	9.5718	0.6060	0.4374
		TOPIC	7	9.4555	0.7008	0.3556
	LogTF-IDF	RPC	7	9.4280	0.6213	0.4601
		En-LDA	19	9.4052	0.5095	0.5345
		TOPIC	11	9.4108	0.6016	0.4895
	LogTF-Entr	RPC	3	10.1156	0.7787	0.2142
		En-LDA	5	10.1365	0.8172	0.1762
		TOPIC	7	10.2031	0.8751	0.1258
	Boolean-TF	RPC	7	6.6159	0.3956	0.8756
		En-LDA	20	7.3124	0.5366	0.7986
		TOPIC	18	6.5699	0.3205	1.3262
D3	TF-IDF	RPC	5	7.5937	0.6728	0.3924
		En-LDA	20	7.3124	0.5366	0.7986
		TOPIC	9	7.4389	0.5966	0.5586
	TF-Entr	RPC	7	8.6670	-0.586	1.9340
		En-LDA	17	8.7483	-0.0428	2.6406
		TOPIC	9	8.7109	-0.0811	2.169
	LogTF-IDF	RPC	4	7.8130	0.7350	0.3088
		En-LDA	19	7.5663	0.5688	0.7740
		TOPIC	13	7.5614	0.5989	0.6396
	LogTF-Entr	RPC	13	9.1759	-0.0690	2.5600
		En-LDA	17	9.2942	-1847	2.8132
		TOPIC	5	8.7880	0.0774	1.6090
	Boolean-TF	RPC	6	3.9200	0.5007	0.6562
		En-LDA	20	3.6822	0.2954	1.4030
		TOPIC	16	3.7309	0.3016	1.3112

generally acceptable. Indeed, the timings report the computation of the TOPIC-SIMILARITY curve for all the *K* values in [2,20], and they are actually due to the computational time of the LDA models themselves.

C. Comparison with state-of-the-art approaches

In this Section, a comparison of the results obtained with TOPIC and state-of-the-art techniques (i.e., *RPC* and *En-LDA*) is presented. RPC [8] is a heuristic algorithm evaluating the average perplexity variation of the LDA models to choose the number of topics. EnLDA [7] is instead an entropy based approach which selects the *K* value in order to minimize the overall amount of entropy of the topic modeling. They will be discussed in more detail in Section II.

Table IV shows the results obtained with the state-of-the-art techniques wrt TOPIC and their evaluation using the same metrics as Table III. It is observable that the first number of topics found by TOPIC-SIMILARITY is comparable with the value found by the RPC method (practically equal in D1 using TF-IDF), which tends to find a very small number of topics. Instead, TOPIC is comparable with En-LDA (which tends to create many clusters, sometimes even taking the upper bound of the possible *K* values as the optimal solution) using the last value of K and as weighting function the Boolean-TF$_{glob}$. This weight finds a greater number of topics to describe the corpus, finding a very fine-grained topics model for the dataset.
Figure 5 shows a subset of most representative topic descriptions obtained with the TF-IDF weighting schema and *K* equal to 10. It is easy to interpret and assign a representative label to each topic that actually describes the main categories originally present in the dataset. These topics are also the bigger ones represented in figure 4, meaning that LDA is able to well identify and split the document into coherent and cohesive clusters. The remaining clusters identified by the LDA model includes words describing more specifically a sub-

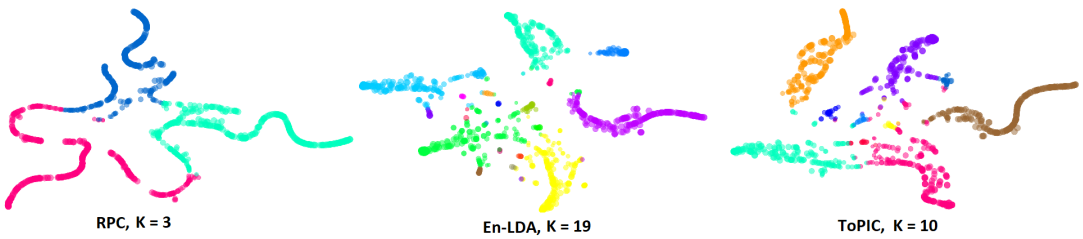

Fig. 4. Comparison of t-SNE representations for dataset D1

Fig. 5. Word cloud representation of a subset of topic, dataset D1, TF-IDF weighting schema, $K = 10$

topic, thus they are not shown in the word cloud representation.

With respect to the computational and time costs, ToPIC outperforms En-LDA. Calculating En-LDA indices is computationally very expensive, and the number of iterations explodes with the growth in documents vocabulary and the cardinality of the corpus. Furthermore En-LDA needs to be computed for all the topics in the given set, having to find the entropy minimum among all the possible K possibilities. RPC, instead, requires a computational time comparable to the one required by ToPIC in the wort case.

V. Conclusion and Future Works

This paper presents a self-tuning data analytics system that effectively mines textual data collections. The proposed framework, ToPIC, includes ad-hoc auto-selection strategies to streamline the analytics process and off-load the parameter tuning from end-user. ToPIC features a distributed implementation in Apache Spark supporting parallel and scalable processing. Possible extensions of the current work are (i) the design of a self-learning strategy able to suggest good configurations which yield higher quality knowledge without performing the analytics task and (ii) the improvement of the characterization of the topics' semantic description to perform better modeling for a given data collection.

References

[1] E. Di Corso, T. Cerquitelli, and F. Ventura, "Self-tuning techniques for large scale cluster analysis on textual data collections," in *Proceedings of the 32nd Annual ACM Symposium on Applied Computing, Marrakesh, Morocco, April 3rd-7th, 2017*, pp. 1–6.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[3] M. Shi, J. Liu, D. Zhou, M. Tang, and B. Cao, "WE-LDA: A word embeddings augmented LDA model for web services clustering," in *2017 IEEE International Conference on Web Services, ICWS 2017, Honolulu, HI, USA, June 25-30, 2017*, 2017, pp. 9–16.

[4] S. Wang, X. Zhao, B. Li, B. Ge, and D. Tang, "Integrating extractive and abstractive models for long text summarization," in *IEEE International BigData Congress*. IEEE, pp. 305–312.

[5] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," in *OSDI'04*, 2004, pp. 10–10.

[6] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *NSDI'12*, 2012, pp. 2–2.

[7] W. Zhang, Y. Cui, and T. Yoshida, "En-lda: An novel approach to automatic bug report assignment with entropy optimized latent dirichlet allocation," *Entropy*, vol. 19, no. 5, p. 173, 2017.

[8] W. Zhao, J. J. Chen, R. Perkins, Z. Liu, W. Ge, Y. Ding, and W. Zou, "A heuristic approach to determine an appropriate number of topics in topic modeling," *BMC bioinformatics*, vol. 16, no. 13, p. S8, 2015.

[9] E. Di Corso, F. Ventura, and T. Cerquitelli, "All in a twitter: Self-tuning strategies for a deeper understanding of a crisis tweet collection," in *IEEE BigData 2017, Boston, MA, USA*, 2017, pp. 3722–3726.

[10] T. Cerquitelli, E. Di Corso, F. Ventura, and S. Chiusano, "Data miners' little helper: Data transformation activity cues for cluster analysis on document collections," in *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics*, ser. WIMS '17. New York, NY, USA: ACM, 2017, pp. 27:1–27:6.

[11] T. Cerquitelli, E. Di Corso, F. Ventura, and S. Chiusano, "Prompting the data transformation activities for cluster analysis on collections of documents," in *Proceedings of SEBD 2017*, 2017, pp. 226–234.

[12] P. Nakov, A. Popova, and P. Mateev, "Weight functions impact on LSA performance," in *EuroConference RANLP'2001 (Recent Advances in NLP*, 2001, pp. 187–193.

[13] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent dirichlet allocation," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 856–864. [Online]. Available: http://papers.nips.cc/paper/3902-online-learning-for-latent-dirichlet-allocation.pdf

[14] E. Spertus, M. Sahami, and O. Buyukkokten, "Evaluating similarity measures: a large-scale study in the orkut social network," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 678–684.

[15] W. B. Towne, C. P. Rosé, and J. D. Herbsleb, "Measuring similarity similarly: Lda and human perception." *ACM TIST*, vol. 8, no. 1, pp. 7–1, 2016.

[16] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008. [Online]. Available: http://www.jmlr.org/papers/v9/vandermaaten08a.html

[17] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, Mar 1986.

[18] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53 – 65, 1987.

[19] A. Spark, "The Apache Spark scalable machine learning library. Available: https://spark.apache.org/mllib/. Last access on February 2018."

[20] T. L. Griffiths and M. Steyvers, "Finding scientific topics," *Proceedings of the National academy of Sciences*, vol. 101, no. suppl 1, pp. 5228–5235, 2004.