Trending Topic Extraction using Topic Models and Biterm Discrimination

Minor Eduardo Quesada Grosso
Universidad de Costa Rica, Escuela de Ciencias de la Computación,
San Pedro, Costa Rica, 11501,
minor.quesada@ucr.ac.cr

and

Edgar Casasola
Universidad de Costa Rica, Escuela de Ciencias de la Computación,
San Pedro, Costa Rica, 11501,
edgar.casasola@ucr.ac.cr

and

Jorge Antonio Leoni de León
Universidad de Costa Rica, Departamento de Lingüística,
San Pedro, Costa Rica, 11501,
antonio.leoni@ucr.ac.cr

Abstract

Mining and exploitation of data in social networks has been the focus of many efforts, but despite the resources and energy invested, still remains a lot for doing given its complexity, which requires the adoption of a multidisciplinary approach . Specifically, on what concerns to this research, the content of the texts published regularly, and at a very rapid pace, at sites of microblogs (eg Twitter.com) can be used to analyze global and local trends. These trends are marked by microblogs emerging topics that are distinguished from others by a sudden and accelerated rate of posts related to the same topic; in other words, by an increment of popularity in relatively short periods, a day or a few hours, for example Wanner et al. . The problem, then, is twofold, first to extract the topics, then to identify which of those topics are trending. A recent solution, known as Bursty Biterm Topic Model (BBTM) is an algorithm for identifying trending topics, with a good level of performance in Twitter, but it requires great amount of computer processing. Hence, this research aims to determine if it is possible to reduce the amount of processing required and getting equally good results. This reduction carry out by a discrimination of co-occurrences of words (biterms) used by BBTM to model trending topics. In contrast to our previous work, in this research, we carry on a more complete and exhaustive set of experiments.

Keywords: Trending topics, topic models, short text, NLP, Natural Language Processing, topic extraction

1 Introduction

With the rise of social networks such as Twitter, many efforts have been made to collect, mine and exploit the information contained in them [1]. The content of posts and comments appearing on these sites can be used to analyze trends in populations overall. The later are marked by emerging issues that are distinguished from others by a sudden and accelerated rate of citations associated with the same topic, in other words, by increasing popularity in relatively short periods, a day or a few hours, for example Wanner et al. [2].

The automatization of trend analysis is of importance for researchers, politicians and companies since these trends manifest thoughts, beliefs, intentions, opinions and wishes of people [3, 4]. For example, it is possible to follow news and observe their evolution over time [2, 5]. Another use is to know what products are popular and the opinions about them [6, 7]. Trend analysis include two subproblems: to identify topics and to identify which of them are trending. The use of short texts, as those present in social networks such as Facebook or Twitter style, add a third problem: The sparsity of words per document makes difficult statistics for trends identification [8]. Therefore, many algorithms to identify emerging topics in social networks often require complicated post- processing

Topic Model Biterm bursty (BBTM) [9] is an algorithm for trending topics identification. This algorithm can identify trending topics in short texts without post- processing. In addition, BBTM gets results above state-of-the-art methods such as Twevent [10], OLDA [11], and UTM [12].

BBTM is a topic model, a widely used technique for topic extraction in text collections [8]. This technique consists in a probabilistic model that finds related terms that identifies topics in a collection of texts. However, to achieve its advantages, instead of using single terms, BBTM associates each word with the others present in the same document (e.g. in Twitter a Tweet is consider a document). These associations of words are named biterms. The use of biterms helps the probabilistic model to get better results with the short text in Twitter, but also increases the amount of data to process. For example, if Twitter gets 73400 different terms in a day and the average number of different terms per document is 5.21, BBTM will have to process the following biterms amount [13]:

$$\frac{246737 \times 5.21(5.21 - 1)}{2} \approx 2705977$$

Experiments carried on a personal computer with 2.6 GHz Intel Core i5 and 8 GB of memory, show to process that quantity of biterms could take about two hours. Beside, most of the memory needed by BBTM is for storage the biterms and their probability to be part of a trending topic.

Therefore, reducing the amount of terms that BBTM has to process is important to decrease the number of biterms and, consequently, reduce the processing time and memory required. These reductions are an significant advantage since in social networks typically a high volume of data is generated in a single day.

Xia et al. [14] proved that it was possible to approximate the most relevant terms for identifying topics in a collection of short texts. The previous implies that there is a set of useless terms equally processed. However, they used BTM [13], an earlier version of BBTM that extracts all topics instead of identifying trends. For this reason, it is necessary to study the effects of discriminate terms on BBTM.

In addition to the possible reduction of computational resources, a decrease of biterms could affect the quality of the results. Our hypothesis is that with less noisy terms, the choice of biterms for each topic would be better.

This article focuses on evaluating the combination of term discrimination and BBTM as a method to reduce the amount of biterm processed. To make this discrimination, we propose to create a graph from the co-occurrence of terms and applying the method introduced in Shetty and Rey [15] to find influential nodes in a graph. Then BBTM would run using mainly the most relevant terms.

The rest of this article is organized as follows: we start with an overview of related work. Then briefly we describe the algorithm BBTM. Following we introduce the proposed method for discrimination of terms . Then the experiments and their results are presented. Finally, conclusions and future work are in the last section.

2 Related work

2.1 Topic models for long texts

BBTM is part of a family of methods called topics models. These methods exploit the semantic structure that is implicit in the texts to model the topics in these texts. Topic models were originally created to extract topics in long texts, such as news articles or texts. For example, Latent Dirichlet Allocation (LDA) [16], is a topic model widely used due to its ability to be extended for getting new features.

2.2 Topic models for short texts

The main problem regarding short texts relates to the sparcity of words. This causes methods for long texts fails due to the unsufficient concurrence of words when they find similarities between texts in order to identify the topics [17]. Hence, there are different solutions to address the short texts processing. One of the approaches to deal with short texts has been the use of external data to enrich their interpretation. For example, in [17] external collections of documents are used to learn topics from them using LDA and then using those topics to help classify short texts.

A similar idea is used in Dual Latent Dirichlet Allocation Model (DLDA) proposed by Jin et al. [18]. This is a version of LDA that learns topics from both collections of long documents and collections of short texts all together, allowing to take advantage of the information in long texts to classify the short texts.

On the other hand, Biterm Topic Model (BTM), presented in Cheng et al. [13], uses a different approach. In order to achieve good results processing short texts, BTM defines a model based on word correlations capable of addressing the problem without pre or post processing. Consequently, other works have been based on BTM extending it to cover different tasks. For example, in Zhu et al. [19] the evolution of topics is modeled through time in microblogs (such as Twitter) using BTM.

2.3 Trending topics extraction

The task of finding trending topics has been addressed in different ways. In Mathioudakis and Koudas [20] a system called TwitterMonitor is proposed. This system detects trending topics identifying emerging keywords in Twitter and grouping them together. In Cataldi et al. [21] trending topics are also detected on Twitter, but modeling the life cycle of the terms to determine the most frequent of these in a specific time interval. Finally, Twevent [10] is a system used to detect events on Twitter. It identifies segments of Tweets frequent in a specific time window.

An important related work is discriminative Biterm Topic Model (d-BTM) [14]. This classifies news on Twitter using BTM as algorithm to identify trending topics and grouping the tweets according to the topics found. However, before running BTM, terms are discriminated to extract those which are representative. Thus, BTM will form the biterms with the most indicative terms of news.

On the other hand, Bursty Biterm Topic Model (BBTM) [9] is a model to identify trending topics also based on BTM. In order to perform this task, BBTM uses data of the explosive popularity of biterms (burstiness) as information during topic modeling process. Compared to previous methods to detect trending topics, BBTM has the following advantages:

1. Because it is based on BTM, it achieves to model short texts effectively overcoming the problem of low density words.
2. Because it incorporates information of the sudden popularity of terms, BBTM can identify trending topics efficiently without heuristics or post processing.

A brief explanation of the general operation of BBTM algorithm is presented bellow.

2.4 Bursty Biterm Topic Model

BBTM models the entire collection of short text as a single document formed by a mixture of topics. Each of these topics is a probability distribution over words. The terms related to trends are emphasized and distributed in different topics, while those terms of common subjects, such as daily life or chatting, are assigned to a single background topic [9]. Figure 1 shows an example of this model, from a collection of twits BBTM extracts the trending topics represented by their keywords, and the rest of standard-use-words are in a unique background topic not included in the results.

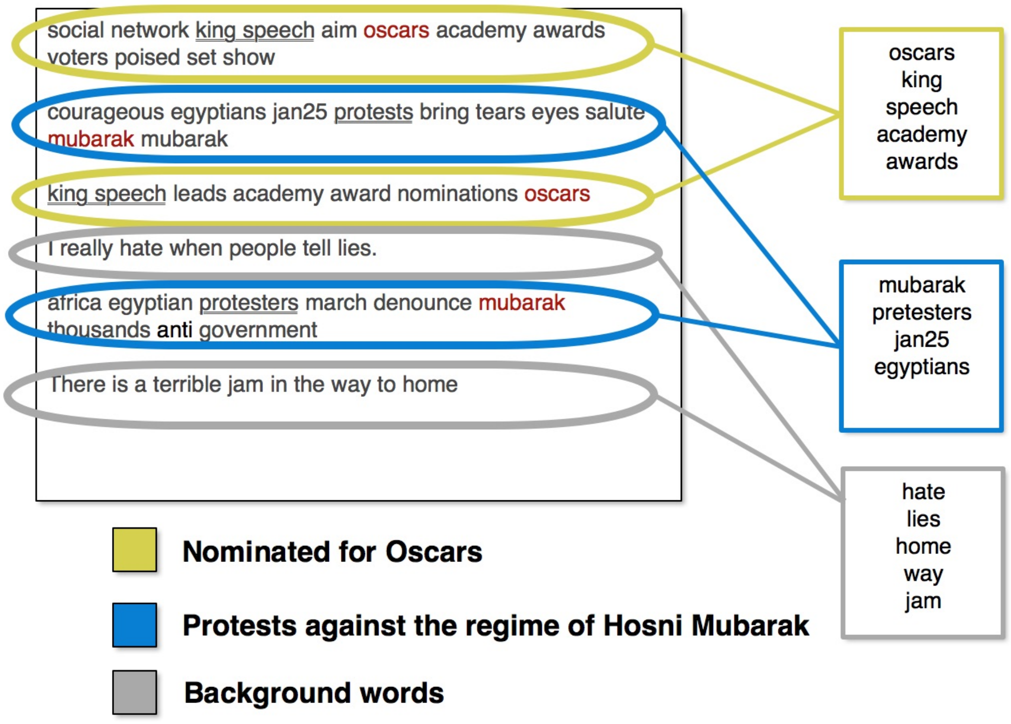
social network king speech aim oscars academy awards voters poised set show
courageous egyptians jan25 protests bring tears eyes salute mubarak mubarak
king speech leads academy award nominations oscars
I really hate when people tell lies.
africa egyptian protesters march denounce mubarak thousands anti government
There is a terrible jam in the way to home
oscars king speech academy awards
mubarak pretesters jan25 egyptians
hate lies home way jam
Nominated for Oscars
Protests against the regime of Hosni Mubarak
Background words

Figure 1: For BBTM in a collection of publications or documents there are several topics and each topic is identify by a set of key terms.

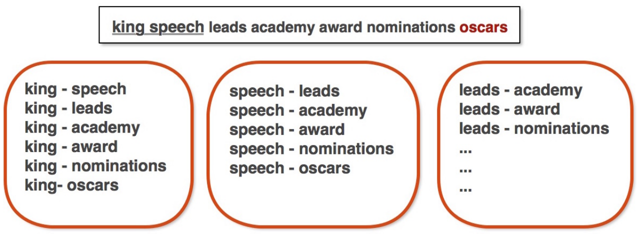

Figure 2: Forming biterms from a Twitter publication

To find the relationship between topics and terms, sufficient samples of word co-occurrence patterns are necessary. Therefore, if we have short texts, and we take each one as an independent document, it will produce the problem of sparse patterns at document-level. BBTM uses the following two strategies to overcome that problem:

1. To use biterms instead of terms: A biterm is an unordered pair of words that co-occurring in the same text. Since a topic is a group of correlated terms, the use of biterms allows to model the co-occurrence of two words explicitly [9, 13]. Figure 2 shows an example of how from a twit biterms are generated.

2. To use the complete collection of texts as a single document: By modeling the co-occurrence of words at corpus-level, BBTM avoids the sparse of patterns problem, in this way the length of the texts does not affect the results [9, 13].

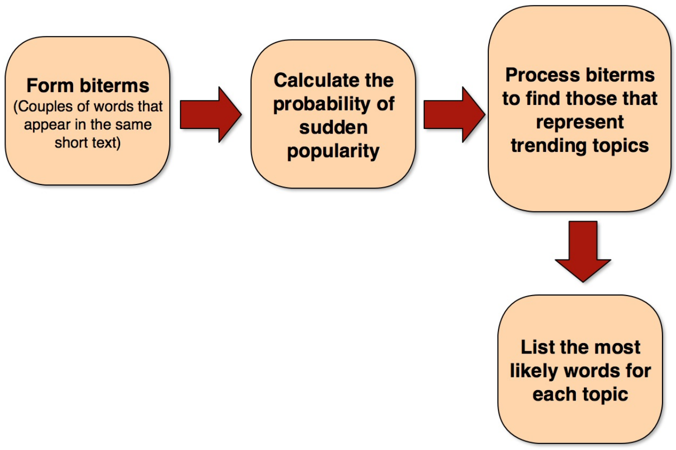
Form biterms
(Couples of words that appear in the same short text)
Calculate the probability of sudden popularity
Process biterms to find those that represent trending topics
List the most likely words for each topic

Figure 3: BBTM steps for extracting trending topics from a collection of short texts.

The steps of BBTM as can be seen in Figure 3 starts with the creation of biterms. The second step is to calculate the probability of each biterm of being relevant to some trending topic. If a biterm has a suddenly popularity (bursty behavior) in contrast to its standard usage in the past, probably that biterm is part of a trend [9]. Thus, the possibility η_b of a biterm generated from a trending topic is calculated as follows:

$$\eta_b = \frac{max(n_b - \bar{n}_b, \epsilon)}{n_b} \tag{1}$$

Where n_b is the biterm frequency within a given period of time (one day for example). $\bar{n}_b$ is the average of the frequencies of that biterm in prior periods (in the previous 10 days for example). The ϵ is to avoid zero probability, according to [9], 0.01 is a good value.

For example, in the day 0 the biterms b_1 b_2 have a frequency of 71 and 15 respectively. In the day 1, the frequencies are 40 and 17, finally for day 2 they are 183 and 28. The average of the frequencies in prior periods are $(71 + 40)/2 = 55.5$ and $(15 + 17)/2 = 16$. Finally, the probabilities of these biterms being part of a trending topic can be calculated:

$$N_{b_1} = \frac{183 - 55.5}{183} = 0.7$$

$$N_{b_1} = \frac{287 - 16}{28} = 0.4$$

After to form biterms and calculate their possibility to be a trend, the third step is to process that data to extract the trending topics. BBTM tries to model how the texts were generated from topics. This generative process is defined as follows [9]:

1. For the collection:

   (a) Draw a trending topic distribution from the collection of texts using a *Dirichlet* probability distribution : θ $Dir(\alpha)$, where α is a hyperparameter of *Dirichlet*.

   (b) Draw a background (no trending) word distribution using a *Dirichlet* probability distribution : ϕ_0 $Dir(\beta)$, where β is a hyperparameter of *Dirichlet*.

2. for each trending topic $z \in [1, K]$:

   (a) Draw a word distribution using a *Dirichlet* probability distribution : ϕ_z $Dir(\beta)$

3. For each biterm $b_i \in B$:

   - Define a binary variable that indicate if the biterm is observed with a normal use or with a trending behavior. The value of this variable is drawn using a *Bernoulli* probability distribution: e_i $Bern(\eta_{b_i})$, where η_{b_i} is the probability defined in Equation 1:

(a) If the biterm has a normal use, $e_i = 0$:
   i. Draw two words $w_{i,1}, w_{i,2}$ using a multinomial probability distribution: $w_{i,1}, w_{i,2}\ Multi(\phi_0)$.

(b) If the biterm has a trending behavior $e_i = 1$:
   i. Draw a trending topic z using a multinomial probability distribution: $z\ Multi(\theta)$:
   ii. Draw two words $w_{i,1}, w_{i,2}$ using a multinomial probability distribution: $w_{i,1}, w_{i,2}\ Multi(\phi_z)$.

Since the *Dirichlet* and multinomial distributions are conjugates to each other, the calculus of their combination gets simpler because the result is another *Dirichlet* distribution. Therefore, the topic z and the biterms $w_{i,1}, w_{i,2}$ are drawn using a multinomial distribution that takes as a parameter the *Dirichlet* used as the prior probability at the beginning. [13].

In the above process the parameters that needs to be estimated are $\theta = \phi_0, \phi_z...\phi_K, \theta$. If the hyperparameters are given, the likelihood of the complete biterm set B is defined as follows [9]:

$$P(B|\alpha, \beta) = \prod_{i=1}^{N_B} \int (\phi_{0,w_{i,1}}\phi_{0,w_{i,2}}(1-\eta_{b_i}) + \sum_{z=1}^{K} \theta_z \phi_{z,w_{i,1}}\phi_{z,w_{i,2}}\eta_{b_i})d\theta \tag{2}$$

The problem with Equation 2 is that results in an intractable integral [9]. For that reason, to approximate the value of θ BBTM use collapsed Gibbs Samplings algorithm.

The gibbs algorithm draw samples from the posterior distribution of the latent variables of topic z and biterm behavior e, sequentially conditioned on the current values of all other variables [9]. This conditional distribution is calculated jointly for both variables:

$$P(e_i = 0|e^{-i}, z^i, B, \alpha, \beta, \eta) \propto (1-\eta_{b_i}) \cdot \frac{(n_{0,w_i,1}^{-i} + \beta)(n_{0,w_i,2}^{-i} + \beta)}{(n_0^{-i} + W\beta)(n_0^{-i} + 1 + \beta)} \tag{3}$$

$$P(e_i = 1, z|e^{-i}, z^i, B, \alpha, \beta, \eta) \propto \eta_{b_i} \cdot \frac{(n_{zb}^{-i} + \alpha)}{(n_b^{-i} + K\alpha)} \cdot \frac{(n_{z,w_i,1}^{-i} + \beta)(n_{z,w_i,2}^{-i} + \beta)}{(n_z^{-i} + W\beta)(n_z^{-i} + 1 + \beta)} \tag{4}$$

Variable	Description
e_i	Indicates the behavior(normal or trendy) of the biterm i
z_i	trending topic assign to the biterm i.
B	Total amount of biterms.
α, β	Hyperparameters of *Dirichlet* distribution.
η	Probability of the biterm to be part of a trending topic.
$n_{0,w}$	Number of times that the word w is assigned with normal use.
n_0	Total of words assigned with normal use.
n_{zb}	Total of biterms assigned to the trending topic z.
n_b	Total of biterms assigned to trending topics.
$n_{z,w}$	Number of times that the word w is assigned to the trending topic z.
n_z	total of words assigned to the trending topic z.
-i	Indicate to ignore the current biterm b_i.

Table 1: conditional distribution for Gibbs sampling variables

The Gibbs sampling algorithm consists in applying Equations 3 and 4 an enough number of repetitions to obtain the necessary data to approximate the parameters $\phi_0, \phi_z...\phi_K, \theta$ using Equations 5 and 5.

$$\phi_{z,w} = \frac{n_{k,w} + \beta}{n_z + W\beta} \tag{5}$$

$$\theta_z = \frac{n_{zb} + \beta}{n_b + W\beta} \tag{6}$$

```
Input  : $K, \alpha, \beta, B$
Output: $\phi_0, \phi_z ... \phi_K, \theta$
Randomly initialize $e$ and $z$
for $iter = 1$ to $N_{iter}$ do
    foreach $b_i(w_{i,1}, w_{i,2}) \in B$ do
        Draw $e$ and $z$ from Equations 3 and 4
        if $e = 0$ then
            Update the value of the variables $n_{0,w_{i,1}}$ and $n_{0,w_{i,2}}$
        else
            Update the value of the variables $n_{zb}$, $n_{z,w_{i,1}}$ and $n_{z,w_{i,2}}$
        end
    end
end
Estimate the parameters $\phi_0, \phi_z ... \phi_K, \theta$ using Equations 5 and 6
```

Algorithm 1: Gibbs sampling algorithm for *BBTM* [9]

Finally, the fourth step (Figure 3) is to list the biterms with more chance to belong to each topic. The term discrimination should assure to keep the words with more probability to be in one of these topics. In this investigation, we evaluate if the results remain equal to the original, they get better with words more relevant to each subject, or they get worse mixing words with a little relation among them.

3 Biterms discrimination method

We propose a method to select the terms more representative to identify trending topics. The idea is that BBTM use the biterms more useful and avoid to process noisy and background biterms. Before to apply the discrimination we need to form the biterms and get their probabilities of sudden popularity, then we form a shorter new list of biterms. Thus, it is an extra step right before the biterm process in BBTM.

The method proposed has three different stages: graph creation, important nodes detection, and selection of terms. During the graph creation stage we link the terms among them. In the important nodes detection stage, the importance or influence of each term into de graph is calculated. Finally, the selection of biterms stage chose the most important terms to be consider like key terms and selecting the biterms related to them.

3.1 Graph creation

We create an undirected graph from the biterms. Each term can be taken as a node with edges or links to terms that forms biterm with it. An example is shown in Figure 4.

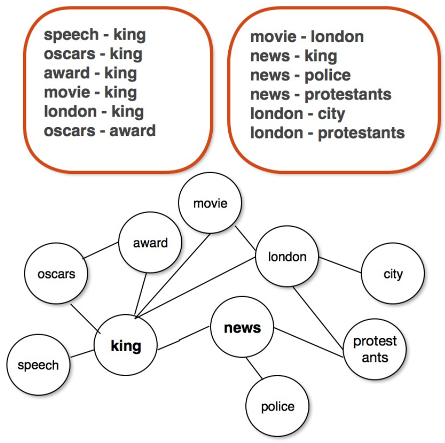

Figure 4: Creation of an undirected graph from a set of biterms.

If a graph is created from all biterms found in a set consisting of a considerable number of texts, the resulting graph will be of considerable size too. Consequently, perform calculations on that graph would be computationally expensive.

For the above reason, it is preferable to construct the graph from biterms with some probability to be trending, in this case those where $\eta_b > \epsilon$. This reduces the graph to a computationally feasible size. Once the graph is built, the next step is to detect influential nodes in the graph.

3.2 Important nodes detection

To select the most influential nodes we chose the algorithm introduced by Shetty and Rey [15]. This technique considers that the most important nodes are those with the most effect the graph entropy when they are eliminated [15]. Therefore, first we must define the calculation of entropy for graph.

In terms of graphs, a high entropy indicates that many edges are equally important, while a low entropy indicates that only a few edges are relevant [22]. It is then possible to calculate the entropy of the graph as follows [22]:

$$H(G) = -\sum_{v \in V} p(v) log(p(v)) \tag{7}$$

Where V is the set of graph nodes. The probability $p(v)$ can be calculated by:

$$p(v) = \frac{|A|}{2|E|} \tag{8}$$

Where $|A|$ is the number of edges of the node and $|E|$ is the total number of edges of the graph.

This measure of entropy is the classical definition of graph complexity based on the symmetries of the graph [23, 24, 25]. Here the value of $p(v)$ is the probability of the node v into the graph. Therefore, this entropy measure is close to the ?information content? in Shannon entropy [25]. We use this definition because we are interested in the structural information content of the graph to know how influential nodes affect to others nodes and the weight of that influence in the whole graph. For that reason, other graph complexity measures, such as *Körner Entropy* [26], are not functional because the structural information is not expressed.

Once the entropy has been defined, the next step is to detect nodes with greater effect on this entropy. To do this, we perform the following algorithm proposed in Shetty and Rey [15]:

```
Input  : Set of nodes $N$.
Output: Importance of each node in $N$.
1) foreach node $N(i) \in N$ do
    a) Compute the entropy of node N (i) by calculating the entropy of the node along with all its
    edges as E (i).
    b)Temporarily drop the node N (i) from the main graph and calculate the entropy of the
    remaining graph as EN (i).
    c) Calculate the importance of the node on the graph with Equation 9.
end
2) Sort nodes by the importance(i) obtained.
```

Algorithm 2: Algorithm to calculate the importance of the nodes of a graph [15]

$$importance = \frac{EN(i)}{\log(\frac{EN(i)}{E(i)})} \tag{9}$$

Equation 9 introduced in [15] is a centrality measure based on the effect of eliminating a node over the entropy of a graph. The basic idea is that most of the information in the graph will be around the key terms of trending topics. This behavior means that it is more probable to reach any of those nodes related to trends, and resting probability to the less relevant nodes, forming in this way, a non-uniform graph. If an influential node is deleted, the rest of the graph become more probable, and consequently a more uniform increasing the entropy. But, if a less relevant node is removed, the graph keep being a non-uniform graph. Thus, a higher affectation on the entropy, the more important is the node.

When we eliminate the node in the step 1b, we also could remove the nodes directly connected with it. In this way, we widen the scope of influence for near nodes [15]. Figure 5 shows an example of this expansion of scope when the node is removed. An entropy model of length 1 means delete only the node, while with length 2 the nodes connected with the node are also removed.

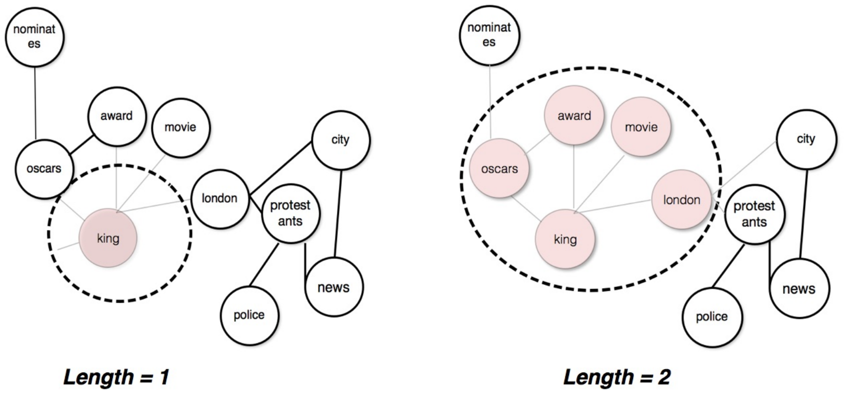

Figure 5: Entropy models with length 1 and 2

3.3 Selection of biterms

The selection of terms is performed based on the degree of importance obtained in the process described above. We define a threshold to identify the most important terms. Any term with equal or greater importance than the threshold is classified as very important. Moreover, those with a lower value are taken as terms with low probability to appear as keywords in any topic. The terms not included in the graph have assigned 0 as importance score. With the selection of important term we can filter the biterms to keep the most relevant. This process is performed as follows:

```
Input  : List of terms $T$ with their importance $I_T$, threshold $\mu$
Output: New list of biterms.
1) foreach term $T_i \in T$ do
    if $I_{T_i} < \mu$ then
        if $I_{T_i} > 0$ then
            a) Find the term $T_{mc}$ with the highest probability of correlation.
            b) Eliminate all biterms associated with $T_i$, except the one formed with $T_{mc}$.
        end
    else
        c) Keep all biterms formed by connecting $T_i$ with the terms $T_j$ where $I_{T_j} >= \mu$ or $I_{T_j} = 0$.
    end
end
```

Algorithm 3: Algorithm to chose the new shorter list of biterms

In summary, for terms with importance below the threshold but different to zero we only keep the biterm formed by connecting the term with the word with the highest probability of correlation. For the rest of terms with importance greater or equal to the threshold, we keep all the biterms formed with it, but those with terms with importance between zero and the threshold.

The probability of correlation can be obtained using the NPMI (Normalized Point Mutual Information) [27]. To avoid the problem of NPMI with pair of words with low frequency, we only take account biterms with frequencies greater than 5. This value was chosen after some preliminary experiments.

The reason for keeping some biterms formed by terms with low importance is because that biterms helps to model correctly the topic. This was proved in [9] when they tried to model using uniquely biterms with high probability to be trend. The results were inferior to those obtained when they used the complete list of biterms. The concept behind the biterm discrimination is to remove a lot of noisy biterms, but at the same time it keeps enough data to model properly the topic.

4 Experimental settings

In this section, we describe our experiments carried on a short text collection of real user data to prove the efficacy of our proposed method. We take BBTM without term discrimination as our baseline. To differentiate both systems, we will refer the original BBTM without term discrimination like single BBTM, and our method like BBTM with term discrimination.

The experiments were executed on a Macbook Pro with Intel Core i5 2.6 Ghz CPU and 8 GB memory. We use the download available original version of BBTM [1]. The term discrimination method was developed like a independent preprocess of BBTM. The implementation was via Java 8.

4.1 Dataset

Experiments were conducted on one week taken from the corpus Tweets2011, TREC collection published in the 2011 microblog track. This portion of the dataset contains 9341618 tweets sampled from Jan. 23 to Jan.29. To reduce the amount of low-quality tweets we apply the same steps as in [9] for BBTM:

1. To delete non-latin characters tweets.
2. To convert all letters to lowercase.
3. To delete stopwords. The corpus includes tweets in several languages, stopwords of English, Spanish, German, Dutch, Indonesian, Portuguese, Brazilian, and French were removed.
4. To delete the 100 most frequent terms, which are common words that are meaningless in Twitter.
5. To delete terms with document frequency less than 10.
6. To filter tweets with length less than 2.
7. To delete duplicate tweets.

After applying the above steps, we left a total of 2421650 tweets and a average of 345950 tweets per day. Next, we take 10 random samples of each day. The size of these subsamples is approximately the half of the complete day sample. Both, BBTM and BBTM with term discrimination were run 10 times. For each one of these runs a different sample was used.

4.2 Evaluation metrics

We evaluate the proposed method of term discrimination with the following metrics:

1. **Recall:** is the fraction of terms that single BBTM retrieves like keyterms of a topic that also was retrieve by BBTM with term discrimination like keyterms of some topic. If A is the result set of keyterms retreived by BBTM with term discrimination, and B is the result set gotten by single BBTM, then the recall is calculated as follows:

$$Recall(A, B) = \frac{|A \cap B|}{|B|} \tag{10}$$

   In this context, recall indicates the percentage to which a topic retrived by BBTM with biterm discrimination is like its equivalent one retrived by single BBTM. We take the 20 first words of both algorithm to evaluate the recall.

2. **Processing time:** We measure the average time processing of Gibbs Sampling for each day. In Addition, we take the average time of the biterm discrimination process.

3. **Coherence:** To evaluate the quality of the topics we use the measure for coherence introduced in [28]. This metric allows to measure if the words selected for a topic appear in the same documents. In this way, if a topic has words not related among them, its coherence will low. This measure is calculated as follows:

$$C(t; V^{(t)}) = \sum_{m=2}^{M} \sum_{l=1}^{m-1} log(\frac{D(w_m, w_l) + 1}{D(w_l)}) \tag{11}$$

   Where $V^{(t)} = (w_1, ..., w_M)$ is the list of the M words that forms the topic t. $D(w)$ document frequency of the word w , y $D(w_1, w_2)$ co-document frequency of the words w_1 y w_2 w_1 .

[1]https://github.com/xiaohuiyan/BurstyBTM

Day	Min	Average	Max
1	5%	41.07%	100%
2	5%	47.6%	100%
3	5%	45.37%	100%
4	5%	47.92%	100%
5	5%	45.3%	100%
6	0%	41.82%	95%

Table 2: Maximum recall average obtained for topics in each day with threshold = -1

Results of our proposed method and our baseline are given independently. Consequently, it is not possible to know with certainty which topics are the equivalents between both results. For this reason, we apply Jaccard index, Precision, Recall against all the resulting topics and reporting the maximum value found. The latter represents the best try to get the same topic that single BBTM.

4.3 Parameters setting

BBTM needs three main parameters to work, two hyperparameters α, β and the number of topics K. We use the same setting of values employed in BBTM [9]: $\alpha = 50/K and \beta = 0.01$. In our experiment, the value of K is fixed to 20 (In this investigation we are not evaluating the behavior on different values of K).

The time slice is set to a day. For each day, the Gibbs sampling algorithm was run for 100 iterations. To define the threshold to select the important keyterms during the term discrimination process we apply standardization. The values of the threshold used are -1, -0.5 and 0.

5 Discussion and results

5.1 Recall

First, we measure how well our method could retrieve similar results than single BBTM. The values of maximum average recall obtain for each threshold tested are shown in Figures 6, 7, 8, and in the tables 2,3,4:

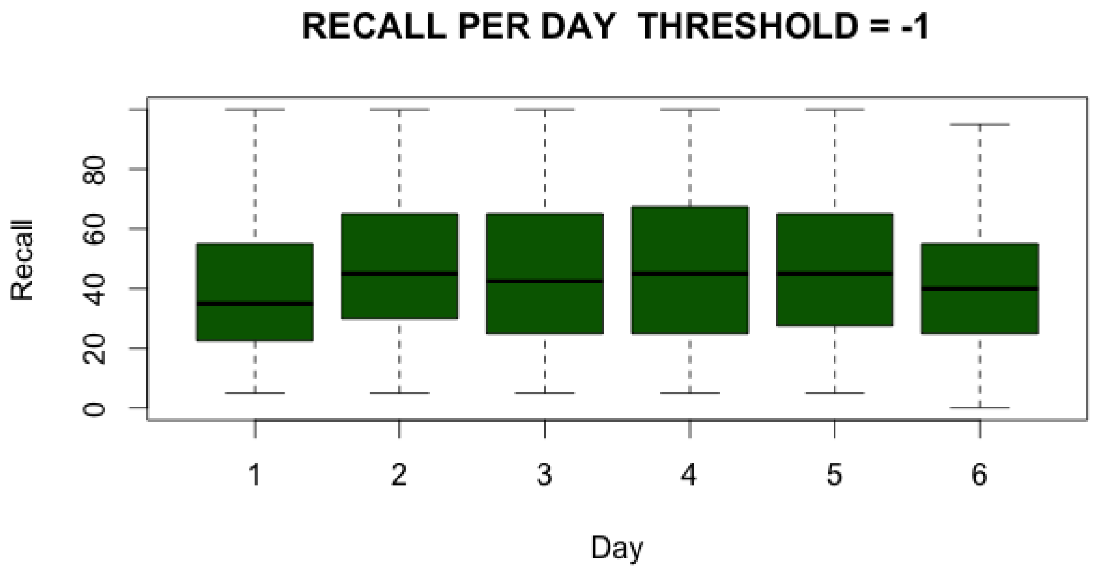

Figure 6: Maximum recall average obtained for topics in each day with threshold = -1

Day	Min	Average	Max
1	0%	38.72%	95%
2	5%	42.55%	95%
3	5%	40.82%	95%
4	0%	40.27%	95%
5	5%	41.2%	95%
6	5%	34.92%	95%

Table 3: Maximum recall average obtained for topics in each day with threshold = -0.5

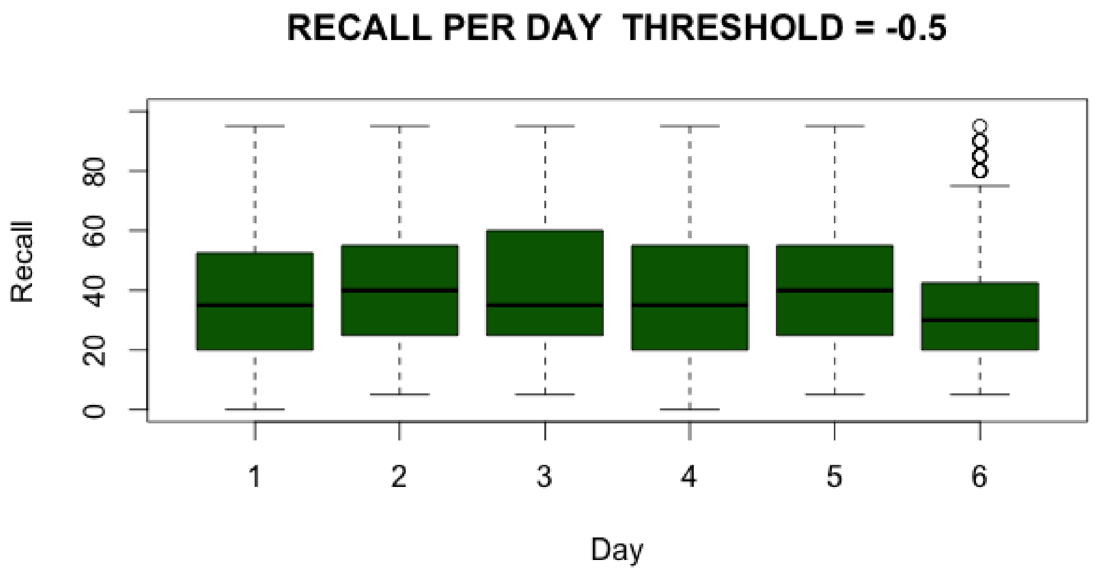

Figure 7: Maximum recall average obtained for topics in each day with threshold = -0.5

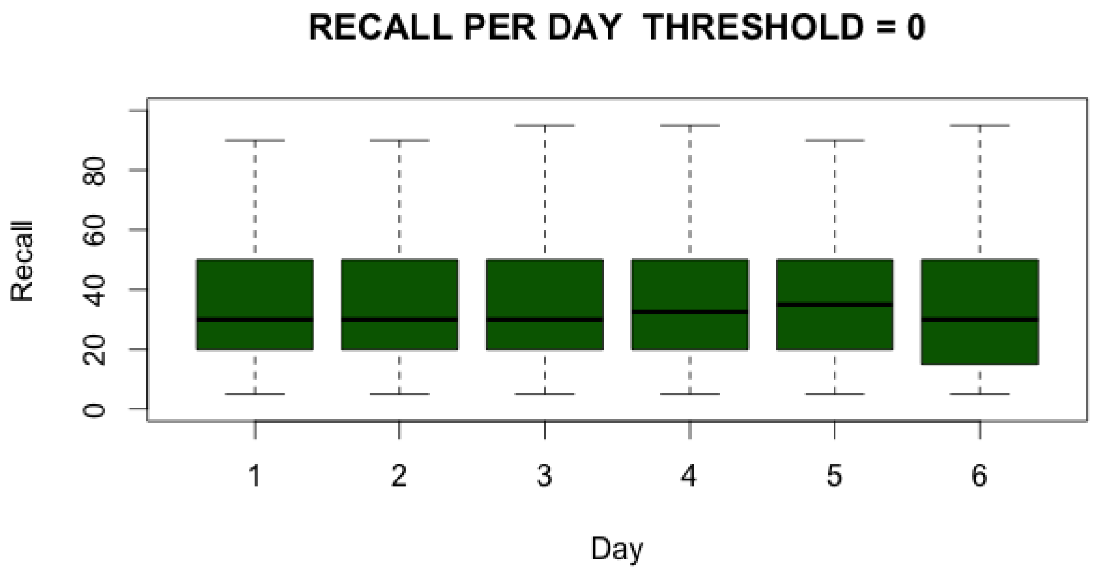

Figure 8: Maximum recall average obtained for topics in each day with threshold = 0

Day	Min	Average	Max
1	5%	37.9%	90%
2	5%	36.2%	90%
3	5%	36.12%	95%
4	5%	36.9%	95%
5	5%	37.42%	90%
6	5%	36.12%	95%

Table 4: Maximum recall average obtained for topics in each day with threshold = 0

Process	Original	-1	-0.5	0
Biterms discrimination	-	1.133		
Day 1	6.949	4.196	2.658	1.908
Day 2	6.916	4.339	2.631	1.944
Day 3	7.546	4.639	3.108	2.290
Day 4	8.090	4.879	3.037	2.567
Day 5	7.522	4.377	2.770	2.184
Day 6	6.218	3.439	2.008	1.941
Total	43.2414	27.002	17.346	13.967

Table 5: Average processing time per day in minutes using different values of threshold.

5.2 Processing time

We compared the processing time of single BBTM and BBTM with biterm discrimination using different values of threshold. Table 5 shows the average processing time per day in minutes.

5.3 Coherence

Table 6 shows the Average coherence per day using different values of threshold:

Day	Original	-1	-0.5	0
1	0.67	0.68	0.64	0.61
2	0.68	0.68	0.66	0.63
3	0.69	0.66	0.63	0.58
4	0.65	0.65	0.61	0.58
5	0.70	0.68	0.65	0.62
6	0.71	0.71	0.62	0.60

Table 6: Average coherence per day.

6 Conclusions and Future Work

We developed a method that can obtain trending topics using less biterms than the original version of BBTM. According to the results presented in section 5, our method, in many cases, is capable of selecting various of the same words to describing each topic. In addition, as it can see in Table 6, the quality of the topics is similar. This goal means that it is possible to reduce the amount of process needed for a trending topic analysis carried on by a topic model like BBTM.

For future work, we would like to try different values of thresholds to separate important keyterms. Also it would be interesting to request different amounts of topics and exploring the behavior our method. Finally, it would be significant to use another technique for term discrimination and compare it with the one used in this work, for example TextRank.

Acknowledgment

We want to express our deep gratitude to Ericka Vargas Castro and Mercedes Villalobos Cardozo for their help during the preparation of this document, and also to the Vicerrectoría de Investigación, Centro de Investigaciones en Tecnologías de la Información y Comunicación (CITIC), the Instituto de Investigaciones Lingüísticas de la Universidad de Costa Rica and Ministerio de Ciencia, Tecnología y Telecomunicaciones

(MICITT) de la República de Costa Rica for the support provided to the research projects N° 745-B4048 and B6175.

References

[1] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining text data*. Springer, 2012, pp. 415–463. [Online]. Available: http://dx.doi.org/10.1007/978-1-4614-3223-4_13

[2] F. Wanner, A. Stoffel, D. Jäckle, B. C. Kwon, A. Weiler, D. A. Keim, K. E. Isaacs, A. Giménez, I. Jusufi, T. Gamblin, and Others, "State-of-the-art report of visual analysis for event detection in text data streams," *Computer Graphics Forum*, vol. 33, no. 3, pp. 1–15, 2014. [Online]. Available: http://dx.doi.org/10.2312/eurovisstar.20141176

[3] M. Injadat, F. Salo, and A. B. Nassif, "Data Mining Techniques in Social Media: A Survey Data Mining Techniques in Social Media: A Survey," *Neurocomputing*, no. I, pp. 1–17, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2016.06.045

[4] L. M. Aiello, G. Petkos, C. Martin, D. Corney, S. Papadopoulos, R. Skraba, A. Goker, I. Kompatsiaris, and A. Jaimes, "Sensing trending topics in twitter," *IEEE Transactions on Multimedia*, vol. 15, no. 6, pp. 1268–1282, 2013. [Online]. Available: http://dx.doi.org/10.1109/TMM.2013.2265080

[5] F. Zarrinkalam and E. Bagheri, "Event Identification in Social Networks," *ArXiv e-prints*, jun 2016.

[6] S. Tuarob and C. S. Tucker, "A product feature inference model for mining implicit customer preferences within large scale social media networks," *ASME IDETC/CIE*, vol. 15, 2015. [Online]. Available: http://dx.doi.org/10.1115/DETC2015-47225

[7] M. Golfarelli, "Social business intelligence: OLAP applied to user generated contents," in *2014 11th International Conference on Wireless Information Netoworks and Systems (WINSYS).* IEEE, 2014, pp. IS—-11.

[8] B. Liu, "Sentiment Analysis and Opinion Mining," *Synthesis Lectures on Human Language Technologies*, vol. 5, no. 1, pp. 1–167, may 2012. [Online]. Available: http://dx.doi.org/10.2200/S00416ED1V01Y201204HLT016

[9] X. Yan, J. Guo, Y. Lan, J. Xu, and X. Cheng, "A Probabilistic Model for Bursty Topic Discovery in Microblogs," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, no. 6, 2015, pp. 353–359.

[10] C. Li, A. Sun, and A. Datta, "Twevent: Segment-based Event Detection from Tweets," *Proceedings of the 21st ACM international conference on Information and knowledge management*, pp. 155–164, 2012. [Online]. Available: http://dx.doi.org/10.1145/2396761.2396785

[11] J. H. Lau, N. Collier, and T. Baldwin, "On-line Trend Analysis with Topic Models:\# twitter Trends Detection Topic Model Online." in *COLING*, 2012, pp. 1519–1534.

[12] H. Yin, B. Cui, H. Lu, Y. Huang, and J. Yao, "A unified model for stable and temporal topic detection from social media data," in *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, apr 2013, pp. 661–672. [Online]. Available: http://dx.doi.org/10.1109/ICDE.2013.6544864

[13] X. Cheng, X. Yan, Y. Lan, and J. Guo, "BTM: Topic Modeling over Short Texts," *Knowledge and Data Engineering, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2014. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2014.2313872

[14] Y. Xia, N. Tang, A. Hussain, and E. Cambria, "Discriminative Bi-Term Topic Model for Headline-Based Social News Clustering," in *Florida Artificial Intelligence Research Society Conference*, 2015, pp. 311–316. [Online]. Available: http://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS15/paper/view/10428

[15] J. Shetty and J. Adibi, "Discovering important nodes through graph entropy the case of enron email database," in *Proceedings of the 3rd international workshop on Link discovery.* ACM, 2005, pp. 74–81. [Online]. Available: http://dx.doi.org/10.1145/1134271.1134282

[16] 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, mar 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944919.944937

[17] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi, "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.* ACM, 2008, pp. 91–100. [Online]. Available: http://dx.doi.org/10.1145/1367497.1367510

[18] O. Jin, N. N. Liu, K. Zhao, Y. Yu, and Q. Yang, "Transferring Topical Knowledge from Auxiliary Long Texts for Short Text Clustering," in *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, ser. CIKM '11. New York, NY, USA: ACM, 2011, pp. 775–784. [Online]. Available: http://dx.doi.org/10.1145/2063576.2063689

[19] J. Zhu, X. Li, M. Peng, J. Huang, and T. Qian, "Coherent Topic Hierarchy: A Strategy for Topic Evolutionary Analysis on Microblog Feeds," in *Web-Age Information Management*, Y. Li, Jian and Sun, Ed. Springer International Publishing, 2015, vol. 9098, no. 61472291, pp. 70–82. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-21042-1_6

[20] M. Mathioudakis and N. Koudas, "Twittermonitor: trend detection over the twitter stream," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data.* ACM, 2010, pp. 1155–1158. [Online]. Available: http://dx.doi.org/10.1145/1807167.1807306

[21] M. Cataldi, L. Di Caro, and C. Schifanella, "Emerging Topic Detection on Twitter Based on Temporal and Social Terms Evaluation," in *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, ser. MDMKDD '10. New York, NY, USA: ACM, 2010, pp. 4:1—-4:10. [Online]. Available: http://dx.doi.org/10.1145/1814245.1814249

[22] R. Navigli and M. Lapata, "Graph connectivity measures for unsupervised word sense disambiguation," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 1683–1688, 2007.

[23] N. Rashevsky, "Life, information theory, and topology," *The bulletin of mathematical biophysics*, vol. 17, no. 3, pp. 229–235, 1955. [Online]. Available: http://dx.doi.org/10.1007/BF02477860

[24] A. Mowshowitz, "Entropy and the complexity of graphs: I. an index of the relative complexity of a graph," *The bulletin of mathematical biophysics*, vol. 30, no. 1, pp. 175–204, 1968. [Online]. Available: http://dx.doi.org/10.1007/BF02476948

[25] A. Mowshowitz and M. Dehmer, "Entropy and the complexity of graphs revisited," *Entropy*, vol. 14, no. 3, pp. 559–570, 2012. [Online]. Available: http://dx.doi.org/10.3390/e14030559

[26] J. Körner, "Coding of an information source having ambiguous alphabet and the entropy of graphs." in *Transactions of the 6-th Prague Conference on Information Theory*, 1973, pp. 411–425.

[27] G. Bouma, "Normalized (Pointwise) Mutual Information in Collocation Extraction," *Proceedings of German Society for Computational Linguistics (GSCL 2009)*, pp. 31–40, 2009.

[28] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum, "Optimizing Semantic Coherence in Topic Models," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, no. 2, 2011, pp. 262–272.