BTM: Topic Modeling over Short Texts

Xueqi Cheng, *Member, IEEE,* Xiaohui Yan, Yanyan Lan, *Member, IEEE,* Jiafeng Guo, *Member, IEEE*

Abstract—Short texts are popular on today's Web, especially with the emergence of social media. Inferring topics from large scale short texts becomes a critical but challenging task for many content analysis tasks. Conventional topic models such as latent Dirichlet allocation (LDA) and probabilistic latent semantic analysis (PLSA) learn topics from document-level word co-occurrences by modeling each document as a mixture of topics, whose inference suffers from the sparsity of word co-occurrence patterns in short texts. In this paper, we propose a novel way for short text topic modeling, referred as *biterm topic model (BTM)*. BTM learns topics by directly modeling the generation of word co-occurrence patterns (i.e., biterms) in the corpus, making the inference effective with the rich corpus-level information. To cope with large scale short text data, we further introduce two online algorithms for BTM for efficient topic learning. Experiments on real-word short text collections show that BTM can discover more prominent and coherent topics, and significantly outperform the state-of-the-art baselines. We also demonstrate the appealing performance of the two online BTM algorithms on both time efficiency and topic learning.

Index Terms—Short Text, Topic Model, Biterm, Online Algorithm, Content Analysis

1 Introduction

Short texts are prevalent on the Web, no matter in traditional Web sites, e.g., Web page titles, text advertisements and image captions, or in emerging social media, e.g., tweets, status messages, and questions in Q&A websites. Unlike traditional normal texts (e.g., news articles and academic papers), short texts, as indicated by the name, typically only include a few words. With the emerging large scale short text datasets, inferring the latent topics from them is important for a wide range of content analysis applications, such as content characterizing [1], [2], [3], user interest profiling [4], and emerging topic detecting [5].

The sparsity of content in short texts brings new challenges to topic modeling. Conventional topic models, such as PLSA [6] and LDA [7], posit that a document is a mixture of topics, where a topic is considered to convey some semantic by a set of correlated words, typically represented as a distribution of words over the vocabulary. Statistical techniques are then utilized to learn the topic components (i.e., topic-word distributions) and mixture coefficients (i.e., topic proportions) of each document. In essence, conventional topic models reveal topics within a text corpus by implicitly capturing the document-level word co-occurrence patterns [8], [9]. Therefore, directly applying these models on short texts will suffer from the severe data sparsity problem (i.e., the sparse word co-occurrence patterns in individual document) [10]. On one hand, the frequency of words in individual short text play less discriminative role than lengthy text, making it hard to infer which words are more correlated in each document [10]. On the other hand, the limited contexts make it more difficult to identify the senses of ambiguous words in short texts.

A simple solution to alleviate the sparsity problem is to aggregate short texts into lengthy pseudo-documents before training a standard topic model. For example, Weng et al. [4] aggregated the tweets published by individual user into one document before training LDA. Besides the user-based aggregation, Hong et al. [10] also aggregated the tweets containing the same word, and showed that topic models trained on these aggregated messages work better than the conventional LDA. However, the effectiveness of such heuristic methods is heavily data-dependent. For example, user information may not be available in some datasets, such as advertisement data. Even if user information is available, e.g., in tweets data, most users only have few tweets that makes the aggregation less effective.

Another way to deal with the problem is to simplify the topic models by adding strong assumptions on short texts. For example, Zhao et al. [2] and Lakkaraju et al. [11] modeled each tweet in the way of mixture of unigrams [12], which assumes a document as a bag of words drawn independently from a single topic. Similar approach can be found in [13], which assumes words in each sentence share a same topic. Although these assumptions may help alleviate the data sparsity problem by simplifying the models, they sacrifice the flexibility to capture multiple topic ingredients in a document. Moreover, they tend to result in peaked posteriors of topics in a document, which makes the model susceptible to overfitting [7].

- *X. Cheng, X. Yan, Y. Lan, J. Guo are with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China. E-mail: {cxq,lanyanyan,guojiafeng}@ict.ac.cn,yanxiaohui@software.ict.ac.cn*
- *Jiafeng Guo is the corresponding author of the paper.*
- *A preliminary version of this paper has been presented at WWW 2013, entitled with "A Biterm Topic Model for Short Texts".*

Unlike these approaches, in this paper, we propose a novel topic model for short texts. The main idea comes from the answers to the following two questions. 1) Since topics are basically groups of correlated words and the correlation is revealed by word co-occurrence patterns in documents, why not explicitly model the word co-occurrence for topic learning? 2) Since topic models on short texts suffer from the problem of severe sparse patterns in individual short document, why not use the rich global word co-occurrence patterns for better revealing topics?

To address these questions, we propose a generative *biterm topic model (BTM)*, which learns topics over short texts by directly modeling the generation of biterms in the whole corpus. Here, a *biterm* is an unordered word-pair co-occurring in a short context (e.g., a small, fixed-size window over a term sequence within a document). BTM posits that the two words in a biterm share the same topic drawn from a mixture of topics over the whole corpus. Here a topic is also represented as a word distribution as conventional topic models. Compared to conventional topic models, the major differences and advantages of BTM lie in that 1) BTM models the word co-occurrence patterns (i.e., biterms) explicitly, rather than implicitly (via document modeling), to enhance topic learning; and 2) BTM uses the aggregated word co-occurrence patterns in the corpus for topic discovering, which avoids the problem of sparse patterns at document-level.

The parameters of BTM can be efficiently estimated by typical algorithms for latent class models, such as Gibbs sampling [14] and variational Bayes [15]. By learning BTM, we can obtain the topic components and a global topic distribution over the corpus, except the topic proportions of individual documents since BTM does not model the document generation process. However, we show that the document-specific topic proportions can be naturally derived based on the learned model in an efficient way.

Another critical issue in short text topic modeling is the scalability of the inference algorithms. As there are millions or even billions of short texts emerging every day, e.g., Twitter tweets or Facebook status messages, developing algorithms that can scale to such massive stream data is a non-trivial problem. Hence, we introduce two online algorithms for BTM, namely online BTM (oBTM) and incremental BTM (iBTM), to speed up the inference of BTM on large datasets. The advantage of the online algorithms is that they only need to store a small fraction of data on the fly for model update, which saves both time and memory cost. Specifically, oBTM runs a batch Gibbs sampler over the biterms in a time slice (e.g., a day) conditioning on the statistics of samples collected in previous time slices, while iBTM updates the model parameters instantly as long as a new biterm is observed.

To measure the performance of BTM, we conducted extensive experiments on three real-world short text collections, i.e., two medium-sized datasets from Twitter and a Q&A website, and a much larger Weibo[1] collection including more than 150 million documents and 9 million distinct terms. Experimental results show that 1) BTM can discover more prominent and coherent topics than the state-of-the-art competitors [10]. When applying the learned topic proportions of documents in short text classification task, we also found that BTM can infer significantly better topic proportions than the baselines. 2) Compared to the batch BTM, the two online algorithms of BTM are much more efficient and comparably effective. Moreover, they substantially outperform the online LDA proposed by Canni et al. [16] in terms of effectiveness. Besides, we also show the online algorithms are capable of capturing the evolution of topics in short text streams.

The rest of the paper is organized as follows. In Section 2, we give a brief survey of related work. Section 3 introduces our model for short text topic modeling, and we discuss its batch implementation in Section 4. Section 5 shows how to infer the topics of a document, and Section 6 presents the two online algorithms for BTM. Experimental results are presented in Section 7. Finally, conclusions are made in the last section.

2 Related Work

In this section, we briefly summarize the related work from the following two perspectives: topic models on normal texts, and that on short texts.

2.1 Topic Models on Normal texts

Topic models are widely used to uncover the latent semantic structure from text corpus. The effort of mining the semantic structure in a text collection can be dated from latent semantic analysis (LSA) [17], which employs the singular value decomposition to project documents into a lower dimensional space, called latent semantic space. Probabilistic latent semantic analysis (PLSA) [6] improves LSA with a sound probabilistic model based on a mixture decomposition derived from a latent class model. In PLSA, a document is represented as a mixture of topics, while a topic is a probability distribution over words. Extending PLSA, Latent Dirichlet Allocation (LDA) [7] adds Dirichlet priors for the document-specific topic mixtures, making it possible to generate unseen documents. Due to its nice generalization ability and extensibility, LDA has achieved huge success in text mining.

In the last decade, topic models have been extensively studied. Many complicated variants and extensions of the standard LDA model have been proposed, which can be found in the comprehensive survey [18].

1. Weibo.com is a popular microblog website in China.

Here we only list some work closely related to us. Wallach [19] proposed the bigram topic model extending LDA by incorporating bigram statistics into topic modeling, but its detail is quite different from ours. The bigram topic model aims to capture ordinal dependencies between words (in normal texts) by exploiting document-level sequential patterns, while our model is designed specifically for short texts and tries to capture the semantic dependencies between words by exploiting corpus-level word co-occurrence patterns. Besides, two recently proposed models, i.e., the regularized topic model [20] and the generalized Pólya model [21], share the same idea of utilizing word co-occurrence (i.e., biterm) statistics to enhance topic learning. However, both of them only use word co-occurrence information as prior to guide the generation of words, rather than directly modeling the co-occurrences. Above all, all these models only deal with normal texts without considering the specificity of short texts.

2.2 Topic Models on Short Texts

Early studies mainly focused on exploiting external knowledge to enrich the representation of short texts. For example, Phan et al.[22] inferred the topics of short texts based on a conventional topic model estimated on another large scale dataset for short text classification. Jin et al.[23] proposed a model based on LDA that jointly learns topics over short texts and related long texts. It is expected to leverage the topical knowledge learned from long texts to help the topic learning task over short texts. However, these methods are only effective when the auxiliary data are closely related to the original data. Sometimes, finding such auxiliary data may be expensive or even impossible. In contrast, our model only relies on statistics of word co-occurrences within the corpus, which is complementary to the above ones. Hence, it is promising to combine them together in future work.

With the emergence of social media in recent years, topic models have been utilized for social media content analysis in various tasks, such as content characterizing [1], [2], event tracking [5], content recommendation [24], [25], and influential users prediction [4]. However, due to the lack of specific topic models for short texts, some researchers directly applied conventional (or slightly modified) topic models [1], [26]. Some others tried to aggregate short texts into lengthy pseudo-documents based on some additional information, and then train conventional topic models [4], [2]. Hong et al. [10] made a comprehensive empirical study of topic modeling in Twitter, and suggested that new topic models for short texts are in demand.

In our previous works, we found the global word co-occurrences is helpful for short text clustering [27] and topic learning [28], [29]. This paper extends our previous conference article [29] with the following improvements. 1) We introduce two online algorithms for BTM to handle large scale datasets. 2) The capability of the two online algorithms is empirically verified. 3) More comprehensive experiments were conducted, and new findings are reported.

3 Biterm Topic Model

In most topic models, topics are represented as groups of correlated words with the correlation basically revealed by word co-occurrence patterns in documents. For example, once observing the words "ipad" and "iphone" frequently co-occurring with each other, one can tell that they have close senses and possibly belong to a same topic, even though he/she doesn't know the exact meaning of them. Conventional topic models exploit word co-occurrence patterns to reveal the latent semantic structure of a corpus in an implicit way by modeling the generation of words in each document. These approaches are sensitive to the shortness of documents since the word co-occurrence patterns in a single short document are sparse and not reliable. Instead, if we aggregate all the word co-occurrence patterns in the corpus, their frequencies are more stable and more clearly reveals the correlation between the words. With this idea, we developed the biterm topic model, which takes a novel way to reveal the latent topic components in a corpus by directly modeling the generation of word co-occurrence patterns.

3.1 Biterm Extraction

Before we detail the model, we first introduce the notation of "biterm", which denotes an unordered word pair co-occurring in a short context (i.e., an instance of word co-occurrence pattern). Here a short context refers to a small, fixed-size window over a term sequence. In short texts with limited document length, such as tweets and text messages, we can simply take each document as an individual context unit. In such case, any two distinct words in a document construct a biterm. For example, a document with three distinct words will generate three biterms:

$$(w_1, w_2, w_3) \Rightarrow \{(w_1, w_2), (w_2, w_3), (w_1, w_3)\},$$

where $(\cdot, \cdot)$ is unordered. After extracting biterms in each document, the whole corpus now turns into a biterm set. The biterm extraction process can be completed via a single scan over the documents.

3.2 Model Description

Unlike most topic models that learn the latent topic components in a corpus by modeling the generate of documents, BTM performs this task by modeling the generation of biterms. The key idea is that if two words co-occur more frequently, they are more likely to belong to a same topic. Based on this idea, we

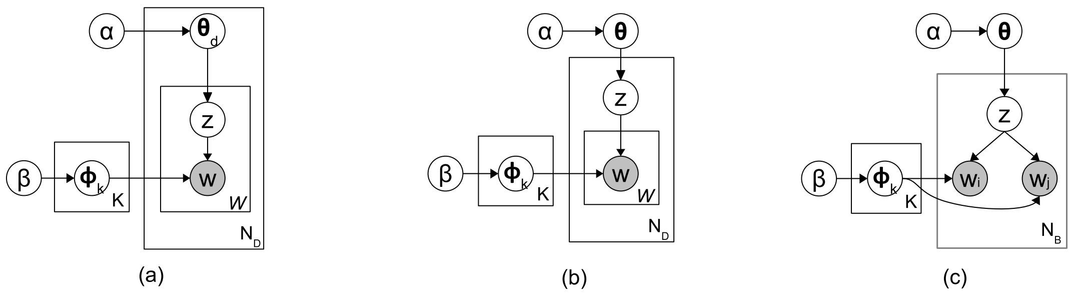


Fig. 1. Graphical representation of (a) LDA, (b) mixture of unigrams, and (c) BTM. Each node in the graph denotes a random variable, where shading represents an observed variable. A plate denote replication of the model within it. The number of replicates is given in the bottom right corner of the plate.

assume that the two words in a biterm are drawn independently from a topic , where a topic is sampled from a topic mixture over the whole corpus.

Given a corpus with N_D documents, suppose it contains N_B biterms $\mathbf{B}=\{b_i\}_{i=1}^{N_B}$ with $b_i=(w_{i,1},w_{i,2})$, and K topics expressed over W unique words in the vocabulary. Let $z\in[1,K]$ be a topic indicator variable, we can represent the prevalence of topics in the corpus (i.e., $P(z)$) by a K-dimensional multinomial distribution $\boldsymbol{\theta}=\{\theta_k\}_{k=1}^K$ with $\theta_k=P(z=k)$ and $\sum_{k=1}^K\theta_k=1$. The word distribution for topics (i.e., $P(w|z)$) can be represented by a $K\times W$ matrix $\mathbf{\Phi}$ where the kth row $\boldsymbol{\phi}_k$ is a W-dimensional multinomial distribution with entry $\phi_{k,w}=P(w|z=k)$ and $\sum_{w=1}^W\phi_{k,w}=1$.

Following the convention of LDA [30], we use symmetric Dirichlet priors for $\boldsymbol{\theta}$ and $\boldsymbol{\phi}_k$ with single-valued hyper-parameters α and β, respectively. Formally, the generative process of BTM is described as follows.

1) Draw $\boldsymbol{\theta}\sim\mathrm{Dirichlet}(\alpha)$
2) For each topic $k\in[1,K]$
   a) draw $\boldsymbol{\phi}_k\sim\mathrm{Dirichlet}(\beta)$
3) For each biterm $b_i\in\mathbf{B}$
   a) draw $z_i\sim\mathrm{Multinomial}(\boldsymbol{\theta})$
   b) draw $w_{i,1},w_{i,2}\sim\mathrm{Multinomial}(\boldsymbol{\phi}_{z_i})$

Its graphical representation is shown in Figure 1(c). Note that we assume that the biterms are generated independently for simplicity.

Following the above procedure, we can write the probability of biterm b_i conditioned on the model parameters $\boldsymbol{\theta}$ and $\mathbf{\Phi}$:

$$\begin{aligned}P(b_i|\boldsymbol{\theta},\mathbf{\Phi}) &= \sum_{k=1}^K P(w_{i,1},w_{i,2},z_i=k|\boldsymbol{\theta},\mathbf{\Phi}).\\ &= \sum_{k=1}^K P(z_i=k|\theta_k)P(w_{i,1}|z_i=k,\phi_{k,w_{i,1}})\cdot\\ &\quad P(w_{i,2}|z_i=k,\phi_{k,w_{i,2}})\\ &= \sum_{k=1}^K\theta_k\phi_{k,w_{i,1}}\phi_{k,w_{i,2}} \qquad (1)\end{aligned}$$

Given the hyperparameters α and β, we can obtain the probability of b_i by integrating over $\boldsymbol{\theta}$ and $\mathbf{\Phi}$:

$$P(b_i|\alpha,\beta)=\int\int\sum_{k=1}^K\theta_k\phi_{k,w_{i,1}}\phi_{k,w_{i,2}}\mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{\Phi} \qquad (2)$$

Taking the product of the probability of single biterms, we obtain the likelihood of the whole corpus:

$$P(\mathbf{B}|\alpha,\beta)=\prod_{i=1}^{N_B}\int\int\sum_{k=1}^K\theta_k\phi_{k,w_{i,1}}\phi_{k,w_{i,2}}\mathrm{d}\boldsymbol{\theta}\mathrm{d}\mathbf{\Phi} \qquad (3)$$

3.3 Model Comparison

For better understanding the uniqueness of BTM, we compare it with two typical models for topic learning, i.e., LDA [7] and mixture of unigrams [12]. In literature, both LDA and mixture of unigram have been employed for topic discovering over short texts [1], [2], [31], [26]. Figure 1 shows the graphical representation of the three models.

LDA, illustrated in Figure 1(a), models the generation of a document d as follows: For each word in d, we first draw a topic z from the document-specific topic distribution $\boldsymbol{\theta}_d$, then draw a word w from topic z. From this figure, we can see that the topic z of word w depends on the other words in the same document through sharing the topic distribution $\boldsymbol{\theta}_d$. Hence, LDA excessively relies on the document-level context for the inference of z and $\boldsymbol{\theta}_d$. It makes LDA susceptible to the data sparsity problem when documents are short, resulting in poor estimation of z and $\boldsymbol{\theta}_d$, in turn, hurting the learning of the topic-word distributions $\mathbf{\Phi}$.

Mixture of unigrams, illustrated in Figure 1(b), also models the generation of each document, but in a different way. It assumes that all the words in a document share a same topic z, while z is sampled from a global topic distribution $\boldsymbol{\theta}$. In other words, it models the whole corpus, rather than a document, as a mixture of topics. By Leveraging the information of the whole corpus, it alleviates the sparsity problem in topic inference over short texts. However, the constraint that a document has a single topic is too strict (as we know that even a short document may contain multiple topics), which prevents it from modeling fine topics in documents.

In a word, the major trouble of LDA and mixture of unigrams lies in modeling the short documents improperly. For such extremely sparse data, it is difficult to directly model and infer the latent topics in single short documents. However, we argue that it is not necessary to model documents for topic discovering in a corpus. BTM, illustrated in Figure 1(c), just chooses another way to discover topics by modeling the generation of biterms, rather than documents. Compared to LDA, BTM avoids the data sparsity problem by learning a global topic distribution $\boldsymbol{\theta}$, as mixture of unigrams does. Meanwhile, by breaking each document into biterms, and assigning a topic for every biterm, BTM allows a document (with multiple biterms) be able to exhibit multiple topics, which surmounts the defect of mixture of unigrams.

4 Parameter Estimation

In this section, we describe the algorithm to estimate the parameters $\{\boldsymbol{\Phi}, \boldsymbol{\theta}\}$ in BTM, and compare its complexity with LDA.

4.1 Gibbs Sampling Algorithm

Similar to LDA, it is intractable to exactly solve the coupled parameters $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ by maximizing the likelihood in Eq. (3). Following [30], we conduct approximate inference for $\boldsymbol{\theta}$ and $\boldsymbol{\Phi}$ using Gibbs sampling [14], which estimates the parameters using samples drawn from the posterior distributions of latent variables sequentially conditioned on the current values of all other variables and the data.

In the setting of BTM, there are three types of variables (i.e., the topic assignments of $\mathbf{z}$, the multinomial distribution parameters $\boldsymbol{\Phi}$ and $\boldsymbol{\theta}$) to be estimated. But using the technique of collapsed Gibbs sampling [32], $\boldsymbol{\Phi}$ and $\boldsymbol{\theta}$ can be integrated out due to the use of conjugate priors. Thus, for biterm b_i, we only need to sample its topic z_i according the following conditional distribution (the derivation is provided in the supplemental material):

$$P(z_i = k|\mathbf{z}_{-i}, \mathbf{B}) \propto (n_{-i,k} + \alpha)\frac{(n_{-i,w_{i,1}|k} + \beta)(n_{-i,w_{i,2}|k} + \beta)}{(n_{-i,\cdot|k} + W\beta + 1)(n_{-i,\cdot|k} + W\beta)}, \tag{4}$$

where $\mathbf{z}_{-i}$ denotes the topic assignments for all biterms except b_i, $n_{-i,k}$ is the number of biterms assigned to topic k excluding b_i, $n_{-i,w|k}$ is the number of times word w assigned to topic k excluding b_i, and $n_{-i,\cdot|k} = \sum_{w=1}^{W} n_{-i,w|k}$. The right hand of Eq. (4) is quite intuitive: the first factor is proportional to the probability of topic k in the corpus, and the second part expresses the product of the probabilities of $w_{i,1}$ and $w_{i,2}$ under topic k.

We summarize the overall procedure of Gibbs sampling in Algorithm 1. Firstly, we randomly assign a topic to each biterm as the initial state. In each iteration, we update the topic assignment for each biterm by examining Eq. (4) sequentially. After a sufficient number of iterations, we count the number of biterms in each topic k, denoting by n_k, and the number of times that each word w assigned to topic k, denoting by $n_{w|k}$. These counts are used to estimate $\boldsymbol{\Phi}$ and $\boldsymbol{\theta}$ as follows (the derivation is presented in the supplemental material):

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

$$\theta_k = \frac{n_k + \alpha}{N_B + K\alpha}. \tag{6}$$

Algorithm 1: Gibbs sampling algorithm for BTM

```
Input: topic number $K$, $\alpha$ and $\beta$, biterm set $\mathbf{B}$
Output: $\boldsymbol{\Phi}$, $\boldsymbol{\theta}$
Randomly initialize the topic assignments for all the biterms
for iter = 1 to $N_{iter}$ do
    foreach biterm $b_i = (w_{i,1}, w_{i,2}) \in \mathbf{B}$ do
        Draw topic $k$ from $P(z_i|\mathbf{z}_{-i}, \mathbf{B})$
        Update $n_k$, $n_{w_{i,1}|k}$, and $n_{w_{i,2}|k}$
Compute $\boldsymbol{\Phi}$ by Eq. (5) and $\boldsymbol{\theta}$ by Eq. (6)
```

TABLE 1
Time complexity and the number of in-memory variables in LDA and BTM

method	time complexity	#in-memory variables
LDA	$O(N_{iter}KN_D\bar{l})$	$N_DK + WK + N_D\bar{l}$
BTM	$O(N_{iter}KN_D\bar{l}(\bar{l}-1)/2)$	$K + WK + N_D\bar{l}(\bar{l}-1)/2$

4.2 Complexity Analysis

We now compare the running time and memory requirement of the Gibbs sampling algorithm of BTM with LDA. We list the time complexity and the number of in-memory variables in the Gibbs sampling procedure of LDA and BTM in Table 1, where $\bar{l}$ denote the average number of words.

In detail, the major time consuming part in the two algorithms is the calculation of conditional probability for topic assignment, which requires $O(K)$ time. Remember that LDA draws a topic for each word occurrence, giving an overall time complexity $O(N_{iter}KN_D\bar{l})$. Instead, BTM draws a topic for each biterm, with the total time complexity $O(N_{iter}KN_B)$. Note that a document with $\bar{l}$ distinct words will generate $l(l-1)/2$ biterms, we roughly have[2]

$$N_B \approx \frac{N_D\bar{l}(\bar{l}-1)}{2}.$$

We can see the time complexity of BTM is about $(\bar{l}-1)/2$ times of LDA. For short texts, since the average length of documents are very small, e.g., $\bar{l} = 5.21$ in the Tweets2011 collection, the run-time of BTM is still comparable with LDA.

2. Here we simply assume that the documents have almost the same number of distinct words. It is reasonable for short texts since the documents are very short.

In the two Gibbs sampling algorithms, the variables necessary to be cached are the counts and topic assignments. In LDA, we need to maintain the counts $n_{k|d}$ (the number of words in document d assigned to topic k), $n_{w|k}$ (the times of word w assigned to topic k), and the topic assignment for each word occurrence [33], in total of $N_D K + WK + N_D \bar{l}$ variables, in-memory. In BTM, we need to keep the counts n_k, $n_{w|k}$, and the topic assignment for each biterm, in total of $K + WK + N_B$ variables, in memory. Compared with BTM, we can see that the memory cost of LDA will increase rapidly when N_D and K become large, making it less memory-efficient than BTM. We will further demonstrate it in the Experiment section.

5 Inferring Topics in a Document

Beside learning the topic components (i.e., $\{\phi_k\}_{k=1}^K$), another common task in topic models is to infer the topics in a document, i.e., evaluating the topic posterior $P(z|d)$ for document d. However, as BTM does not model documents, we cannot directly obtain $P(z|d)$ from the estimated model. Fortunately, we can derive the topic proportion of a document via the topics of biterms.

Suppose d contains N_d biterms, $\{b_i^{(d)}\}_{i=1}^{N_d}$, using the chain rule we have

$$P(z|d) = \sum_{i=1}^{N_d} P(z, b_i^{(d)}|d) = \sum_{i=1}^{N_d} P(z|b_i^{(d)}, d) P(b_i^{(d)}|d). \tag{7}$$

Given biterm $b_i^{(d)} = (w_{i,1}^{(d)}, w_{i,2}^{(d)})$, we assume its topic z is conditionally independent of d, i.e., $P(z|b_i^{(d)}, d) = P(z|b_i^{(d)})$. Then, we can simplify the above equation:

$$P(z|d) = \sum_{i=1}^{N_d} P(z|b_i^{(d)}) P(b_i^{(d)}|d). \tag{8}$$

In Eq. (8), $P(z|b_i^{(d)})$ can be calculated via Bayes' formula based on the parameters learned in BTM:

$$P(z = k|b_i^{(d)}) = \frac{\theta_k \phi_{k,w_{i,1}^{(d)}} \phi_{k,w_{i,2}^{(d)}}}{\sum_{k'} \theta_{k'} \phi_{k',w_{i,1}^{(d)}} \phi_{k',w_{i,2}^{(d)}}} \tag{9}$$

Meanwhile, $P(b_i^{(d)}|d)$ can be estimated empirically:

$$P(b_i^{(d)}|d) = \frac{n(b_i^{(d)})}{\sum_{i=1}^{N_d} n(b_i^{(d)})},$$

where $n(b_i^{(d)})$ is the frequency of biterm $b_i^{(d)}$ in d.

6 Online Algorithms for BTM

In real-world applications such as microblog, short texts are often with prohibitively large volume, coming in a stream, and growing rapidly over time. In such case, the batch algorithm is no longer suitable for topic learning. First of all, it is impractical to scan the whole dataset repeatedly due to the limitation of memory. Second, it is desired to keep the model up-to-date when new data arrive continuously. For these reasons, we introduce two online algorithms for BTM, referred as online BTM (oBTM) and incremental BTM (iBTM). The online algorithms only need to store a small fraction of data on the fly for model update, which are much more efficient than the batch algorithm on large scale dataset.

Algorithm 2: Online BTM Algorithm

```
Input: $K, \alpha, \beta, \lambda$, Biterm sets $\mathbf{B}^{(1)}, ..., \mathbf{B}^{(T)}$
Output: $\{\mathbf{\Phi}^{(t)}, \boldsymbol{\theta}^{(t)}\}_{t=1}^T$
Set $\boldsymbol{\alpha}^{(1)} = (\alpha, ..., \alpha)$ and $\{\boldsymbol{\beta}_k^{(1)} = (\beta, ..., \beta)\}_{k=1}^K$
for $t = 1$ to $T$ do
    Randomly assign topics to biterms in $\mathbf{B}^{(t)}$
    for $iter = 1$ to $N_{iter}$ do
        foreach biterm $b_i = (w_{i,1}, w_{i,2}) \in \mathbf{B}^{(t)}$ do
            Draw topic $k$ from Eq. (10)
            Update $n_k^{(t)}$, $n_{w_{i,1}|k}^{(t)}$, and $n_{w_{i,2}|k}^{(t)}$
        Set $\boldsymbol{\alpha}^{(t+1)}$ and $\{\boldsymbol{\beta}_k^{(t+1)}\}_{k=1}^K$ by Eq.(11) and Eq.(12)
    Compute $\mathbf{\Phi}^{(t)}$ by Eq.(5) and $\boldsymbol{\theta}^{(t)}$ by Eq.(6)
```

6.1 Online BTM Algorithm (oBTM)

The oBTM algorithm is inspired by the online LDA algorithm proposed in [34], which assumes documents are divided by time slices (e.g., a day), and the documents are exchangeable in a time slice. The main idea of oBTM is to fit a BTM model over the data in a time slice t and use the counts in current time slice, $n_k^{(t)}$ and $n_{w|k}^{(t)}$, to adjust the Dirichlet hyperparameters for the next time slice. The overall procedure of oBTM is outlined in Algorithm 2.

Before running oBTM, we need to transform documents in time slice t into biterm set $\mathbf{B}^{(t)}$. Let $\boldsymbol{\alpha}^{(t)}$ be the K-dimensional Dirichlet hyperparameters for $\boldsymbol{\theta}^{(t)}$, and $\boldsymbol{\beta}_k^{(t)}$ be the W-dimensional Dirichlet hyperparameters for $\boldsymbol{\phi}_k^{(t)}$. We use symmetric Dirichlet distributions as the initial priors by setting $\boldsymbol{\alpha}^{(1)} = (\alpha, ..., \alpha)$ and $\boldsymbol{\beta}_k^{(1)} = (\beta, ..., \beta)$. Given $\boldsymbol{\alpha}^{(t)}$ and $\{\boldsymbol{\beta}_k^{(t)}\}_{k=1}^K$, we iteratively draw topic assignments for each biterm $b_i \in \mathbf{B}^{(t)}$ according to the conditional distribution:

$$P(z_i = k|\mathbf{z}_{-i}^{(t)}, \mathbf{B}^{(t)}, \boldsymbol{\alpha}^{(t)}, \{\boldsymbol{\beta}_k^{(t)}\}_{k=1}^K) \propto$$
$$(n_{-i,k}^{(t)} + \alpha_k^{(t)}) \frac{(n_{-i,w_i|k}^{(t)} + \beta_{k,w_i}^{(t)})(n_{-i,w_j|k}^{(t)} + \beta_{k,w_j}^{(t)})}{[\sum_{w=1}^W (n_{-i,w|k}^{(t)} + \beta_{k,w}^{(t)}) + 1][\sum_{w=1}^W (n_{-i,w|k}^{(t)} + \beta_{k,w}^{(t)})]} \tag{10}$$

Once iterations completed, we obtain the counts $n_k^{(t)}$ and $n_{w|k}^{(t)}$, and utilize them to adjust the hyperparameters $\boldsymbol{\alpha}^{(t+1)}$ and $\{\boldsymbol{\beta}_k^{(t+1)}\}_{k=1}^K$ for time slice $t+1$ by setting:

$$\alpha_k^{(t+1)} = \alpha_k^{(t)} + \lambda n_k^{(t)}, \tag{11}$$
$$\beta_{k,w}^{(t+1)} = \beta_{k,w}^{(t)} + \lambda n_{w|k}^{(t)}, \tag{12}$$

where $\lambda \in [0, 1]$ is a decay weight. As stated in [35], [34], by virtue of the Dirichlet-multinomial conjugate

Algorithm 3: Incremental BTM Algorithm

```
Input: $K, \alpha, \beta$, Biterm sequence $\mathbf{B} = \{b_1, ..., b_N\}$
Output: $\mathbf{\Phi}, \boldsymbol{\theta}$
for $i = 1$ to $N$ do
    Draw topic $k$ from $P(z_i|\mathbf{z}_{i-1}, \mathbf{B}_i)$
    Update $n_k$ and $n_{w|k}$
    Generate rejuvenation sequence $R(i)$
    for $j \in R(i)$ do
        Draw topic assignment $k'$ from $P(z_j|\mathbf{z}_{-j,i}, \mathbf{B}_i)$
        Update $n_{k'}$ and $n_{w|k'}$
Compute $\mathbf{\Phi}$ by Eq.(5) and $\boldsymbol{\theta}$ by Eq.(6)
```

property the hyperparameters $\alpha_k^{(t)}$ and $\beta_{k,w}^{(t)}$ can be viewed as the counts of prior observations of $n_k^{(t)}$ and $n_{w|k}^{(t)}$, respectively. Therefore, Eqs.(11-12) can be interpreted as taking historical topic assignments as prior observations for the next time slice. Additionally, the decay weight λ controls the strength of influence of historical topic assignments. If $\lambda = 0$, the models trained in different time slices are completely independent; If $0 < \lambda < 1$, the historical influence will decays exponentially with the number of time slices passed; If $\lambda = 1$, the historical counts of topic assignments are simply accumulated without any decay.

By running the batch Gibbs sampler over the data in each time slice sequentially, oBTM is simple and easy to be implemented. However, in some applications, such as real-time topic tracking in microblogs, it is desired to update the model instantly when new documents arrive. In such case, oBTM is incompetent. Hence, we turn to another online algorithm more appropriate for such tasks.

6.2 Incremental BTM Algorithm (iBTM)

iBTM updates the model continuously, i.e., updating the parameters $\mathbf{\Phi}$ and $\boldsymbol{\theta}$ immediately whenever a biterm arrives, via a technique called incremental Gibbs sampler [16]. In detail, when biterm b_i arrives, iBTM updates the model in two steps. First, we draw the topic assignment of b_i from $P(z_i|\mathbf{z}_{i-1}, \mathbf{B}_i)$, where $\mathbf{z}_{i-1} = \{z_j\}_{j=1}^{i-1}$ indicates all the previous topic assignments, and $\mathbf{B}_i = \{b_j\}_{j=1}^{i}$. Second, we randomly choose some previous biterms to construct a biterm sequence, called rejuvenation sequence $R(i)$, to resample their topic assignments. For each biterm $b_j \in R(i)$, we resample its topic assignment z_j from $P(z_j|\mathbf{z}_{-j,i}, \mathbf{B}_i)$. The procedure of iBTM is outlined in Algorithm 3.

A crucial concern of iBTM is how to generate the rejuvenation sequence $R(i)$. First, the length of $R(i)$ makes a trade-off between effectiveness and efficiency. The more biterms rejuvenated, the better approximation of the posterior distribution $P(\mathbf{z}_i|\mathbf{B}_i)$ will be achieved. Particularly, if $R(i)$ is set to $\mathbf{B}_i$, iBTM approaches to the batch BTM algorithm as the number of biterms increases to infinity, since every topic assignment will be resampled infinite times. Second, the choice of $R(i)$ can affect the contribution of biterms received at different time in model update. For instance, one can select entries in $R(i)$ from decayed distributions (e.g., exponential and inverse polynomial distributions [36]) over previous biterms to favor more recent historical data. In this work, $R(i)$ is generated from a uniform distribution over a fixed-size sliding window covering the recent biterms. This approach not only reduces the memory and time cost by storing a small part of historical data, but also makes the model more sensitive to the dynamic changes of topics in data than oBTM, since it updates the model continuously.

TABLE 2
Time complexity and the number of in-memory variables of batch and online BTM algorithms in time slice t

method	time complexity	#in-memory variables
batch BTM	$O(N_{iter}K\|\mathbf{B}^{(1..t)}\|)$	$K + WK + \|\mathbf{B}^{(1..t)}\|$
oBTM	$O(N_{iter}K\|\mathbf{B}^{(t)}\|)$	$K + WK + \|\mathbf{B}^{(t)}\|$
iBTM	$O(K\|\mathbf{B}^{(t)}\| \cdot \|R(i)\|)$	$K + WK + L$

6.3 Complexity Comparison

One major advantage of the online algorithms against the batch algorithm is that they scale well to massive datasets, since they only need to store a small fraction of data for model update. By analyzing the time complexity and memory consumption, we can be more clear about this. To facilitate comparison, we assume the corpus is organized with T time slices, and then estimate the running time and memory requirements for model update in time slice t.

The batch BTM, which needs to run over all the biterms observed up to time slice t (i.e., $\mathbf{B}^{(1..t)} = \mathbf{B}^{(1)} \cup ... \cup \mathbf{B}^{(t)}$), costs time $O(N_{iter}K|\mathbf{B}^{(1..t)}|)$, and has to record $K + WK + |\mathbf{B}^{(1..t)}|$ variables in memory, where $|\cdot|$ denotes the number of elements in a set. Instead, oBTM only iteratively runs over the biterm set $\mathbf{B}^{(t)}$ in the current time slice, with time complexity $O(N_{iter}K|\mathbf{B}^{(t)}|)$ and $K + WK + |\mathbf{B}^{(t)}|$ variables in memory. For iBTM, it runs over $\mathbf{B}^{(t)}$ in a single pass, but along with $|R(i)|$ times of resampling for each biterm. Thus its time complexity is $O(K|\mathbf{B}^{(t)}| \cdot |R(i)|)$, and the number of in-memory variables is $K+WK+L$, where L is the length of the sliding window. Note that $L \ll |\mathbf{B}^{(1..t)}|$ on large datasets.

Table 2 summarizes the time complexity and memory consumption of the batch BTM algorithm, oBTM and iBTM. We can see that the time and memory cost of the batch BTM algorithm increase linearly as t grows. In contrary, the two online BTM algorithms require almost constant time and memory cost to update the model, since the number of biterms in each time slice is often stable. Therefore, the two online algorithms can handle large-scale datasets efficiently by processing the data incrementally.

TABLE 3
Summary of the three short text collections.

Dataset	Question	Tweets2011	Weibo
#doc	189,080	4,230,578	155,617,473
#word	26,565	98,857	187,994
avgDocLen	3.94	5.21	5.87

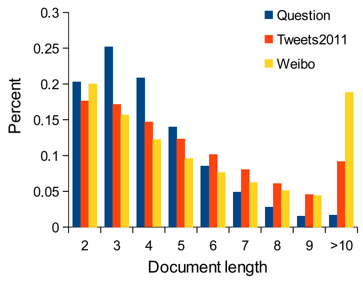


Fig. 2. Document length (i.e., number of words) distribution of the three collections.

7 Experiments

In this section, we empirically evaluate the effectiveness and efficiency of BTM, both the batch and online BTM algorithms.

7.1 Experimental Settings

Datasets. In order to show the effectiveness of our approach over different short text datasets, we use three short text collections for evaluation.

- **Question** collection includes 648,514 questions crawled from a popular Chinese Q&A website[3]. In this collection, each question has a label chosen from 35 categories by its author.
- **Tweets2011** collection is a standard short text collection published in TREC 2011 microblog track[4], which provides approximately 16 million tweets sampled between January 23rd and February 8th, 2011. Besides its content, each tweet includes a user id and a timestamp.
- **Weibo** collection is a subset of microblogs collected from weibo.com between 2011/08/01 and 2012/07/31. The volume of raw data is about 600G.

The raw data of these collections are very noisy. For preprocessing, we removed meaningless words such as stop words, low frequency words, and characters not in Latin or Chinese. To filter out low-quality documents, we removed duplicate documents and documents with a single word. Table 3 lists the number of documents and distinct words, and the average length (i.e., number of words) of documents of the three collections after preprocessing.

We first evaluate the batch BTM on the Question and Tweets2011 collections, since the Weibo collection is so large that the cost of running the batch algorithms is prohibitively expensive. We then use the Tweets2011 and Weibo collections to examine the performance of the two online algorithms, since they have timestamps with the posts, and can be treated as text streams.

Baseline Methods. For batch BTM, we compared it with three typical methods for short text topic modeling nowadays:

- **Mix** denotes the mixture of unigrams model which assumes each document only exhibits a single topic.
- **LDA** denotes the standard LDA model implemented by Gibbs sampling[5].
- **LDA-U** aggregates all the posts of a user into a longer pseudo-document, and then applies LDA on these pseudo-documents.

Besides, we also compared the performance of the two online algorithms of BTM with the online LDA, referred as **iLDA**, a major competitor on large scale datasets. For online LDA, we chose the version implemented by incremental Gibbs sampler [16]. Since its effectiveness and efficiency are as good as, if not better than, other implementations such as particle filter [16] and online variational Bayes [37] in our preliminary experiments. To be fair, all the methods are implemented by Gibbs sampling in C++[6]. All the experiments are carried on a Linux server with Intel Xeon 2.33 GHz CPU and 16G memory.

The parameters α and β were tuned via grid search on the smallest collection (i.e., Question) in our experiments [7]. We found when $\alpha = 0.05$ and $\beta = 0.01$, LDA almost always achieved the best performance in the preceding experiments. For BTM and mixture of unigrams, we found $\alpha = 50/K$ and $\beta = 0.01$ always works well. When comparing with the batch algorithm, we set $\lambda = 1$ in oBTM, so that the data in different time slices contribute equally. For oBTM, the time slices are split by day on the two microblog datasets. To compare the performance of the online algorithms, we set the length of the rejuvenation sequence $|R(i)|$ in iBTM to be N_{iter}, and the sliding window length L to be $|\mathbf{B}^{(1)}|$, so that its time and memory cost will be roughly equal to oBTM. Gibbs sampling was run for 1,000 iterations on the Question and Tweets2011 collections. Considering the Weibo collection is too large, we ran 100 iterations on it.

Measures and Methodology. We aim to evaluate the effectiveness and efficiency of the batch and online BTM algorithms on short texts. Note that the evaluation of effectiveness of a topic model is not a trivial problem. A typical metric is the perplexity or marginal

3. http://zhidao.baidu.com
4. http://trec.nist.gov/data/tweets/
5. http://gibbslda.sourceforge.net/
6. Code of BTM : http://code.google.com/p/btm/
7. Specifically, we first varied the parameters in {0, 0.01, 0.1, 1, 10} to find the best ones, and then fine-tuned the parameters near these values to determine the final values of the parameters.

TABLE 4
PMI-Scores of the batch algorithms. A larger PMI-Score indicates more coherent topics.

	K	50			100		
Collection	Method	Top5	Top10	Top20	Top5	Top10	Top20
Question	LDA	2.15 ± 0.05	1.70 ± 0.03	1.40 ± 0.04	2.16 ± 0.04	1.71 ± 0.03	1.39 ± 0.02
	Mix	2.28 ± 0.06	1.82 ± 0.03	1.43 ± 0.03	2.34 ± 0.05	1.80 ± 0.04	1.40 ± 0.03
	BTM	$\mathbf{2.34 \pm 0.05}$	$\mathbf{1.88 \pm 0.03}$	$\mathbf{1.48 \pm 0.03}$	$\mathbf{2.42 \pm 0.06}$	$\mathbf{1.89 \pm 0.05}$	$\mathbf{1.49 \pm 0.03}$
Tweets2011	LDA	2.61 ± 0.06	1.93 ± 0.04	1.77 ± 0.02	2.64 ± 0.06	2.02 ± 0.04	1.78 ± 0.02
	LDA-U	2.63 ± 0.02	2.14 ± 0.06	1.77 ± 0.02	2.72 ± 0.02	2.20 ± 0.02	1.79 ± 0.01
	Mix	2.72 ± 0.07	2.19 ± 0.03	1.83 ± 0.02	2.85 ± 0.04	2.28 ± 0.02	1.83±0.02
	BTM	$\mathbf{2.74 \pm 0.04}$	$\mathbf{2.26 \pm 0.04}$	$\mathbf{1.86 \pm 0.02}$	$\mathbf{2.88 \pm 0.02}$	$\mathbf{2.33 \pm 0.04}$	$\mathbf{1.87 \pm 0.03}$

likelihood evaluated on a held-out test set [7], [13], [38], but it is not suitable for us for two reasons. First, the marginal likelihoods of LDA and BTM are not comparable, since LDA optimizes the likelihood of word occurrences in documents, while BTM optimizes the likelihood of biterm occurrences in the corpus. Second, these metrics disconnect with our expectations of topic models [18], e.g., the interpretability of topics and usefulness in real applications. It is argued that topic models with better held-out likelihood may infer less semantically meaningful topics [39]. Considering that we are often interested in two parts of the results of topic models, i.e., the topic components and documents' topic proportions, we would like to evaluate the quality of them separately.

In recent years, some automatic evaluation methods are proposed to measure the quality of the topics discovered. One is the *coherence score* [21], which says that a topic is more coherent if the most probable words in it co-occurring more frequently in the corpus. This idea is consistent with the basic assumption of BTM, i.e., words co-occurring more frequently should be more possible to belong to a same topic. Thus it is not surprising that BTM always obtains better coherence scores than the baselines, as shown in our preliminary work [29]. Another popular metric for automatic evaluation is the PMI-Score [40], which measures the coherence of a topic based on pointwise mutual information using large scale text datasets from external sources, e.g., Wikipedia and Baike[8]. Since these external datasets are model-independent, PMI-Score is fair for all the topic models. Therefore, we exploit PMI-Score to verify the topic quality.

Given the T most probable words of a topic k, $(w_1, ..., w_T)$, PMI-Score measures the pairwise association between them:

$$\text{PMI-Score}(k) = \frac{1}{T(T-1)} \sum_{1 \le i < j \le T} \text{PMI}(w_i, w_j),$$

where $\text{PMI}(w_i, w_j) = \log \frac{P(w_i,w_j)}{P(w_i)P(w_j)}$, $P(w_i, w_j)$ and $P(w_i)$ are the probabilities of co-occurring word pair (w_i, w_j) and word w_i estimated empirically from the external datasets, respectively. For evaluation on the Tweets2011 corpus, we compute the PMI-Score using 4M English Wikipedia articles. For the Question and Weibo datasets, we compute the PMI-Score using 5M Chinese Baike articles.

To measure the quality of the documents' topic proportions, we use document classification to see how accurate and discriminative of the learned topical representations from different models are. For each document d, its topical representation is a vector $[P(z=1|d), ..., P(z=K|d)]$[9]. We randomly split the dataset into training and test subsets with the ratio $4:1$, and employed the linear SVM classifier LIBLINEAR[10] for classification with 5-fold cross validation.

Note that in the microblogs datasets, i.e., Tweets2011 and Weibo, there is no category information for documents. Manual labeling might be difficult due to the incomplete and informal content of microblogs. Fortunately, some microblogs are labeled by their authors with hashtags in the form of "#keyword". By investigating the data, we find there are mainly three types of usage of hashtags: (a) marking events or topics; (b) defining the types of content, such as "#ijustsayin", "#quote"; (c) realizing some specified functions, such as "#fb" means importing the tweet to Facebook. We manually chose 50 frequent hashtags in type (a) as class labels, and collect documents with these hashtags for classification. Documents with more than one hashtags are discarded. Table 5 lists the 50 hashtags selected from the Tweets2011 collection.

TABLE 5
Hashtags selected for evaluation from Tweets2011.

jan25	superbowl	sotu	wheniwaslittle	mobsterworld	jobs	
agoodboyfriend	bieberfact	glee	lfc	rhoa	itunes	thegame
celebrity	tcyasi	americanidol	cancer	socialmedia	jerseyshore	
photography	jp6foot7remix	factsaboutboys	meatschool			
libra	android	sagittarius	thissummer	tnfisherman	sagawards	
ausopen	bears	weather	jaejoongday	skins	bfgw	fashion
pandora	realestate	teamautism	travel	nba	football	marketing
design	oscars	food	dating	kindle	snow	obama

8. The most popular Chinese version of Wikipedia: http://baike.baidu.com

9. In Mix, the topic posterior of a document $d = (w_1, ..., w_n)$ can be inferred using Bayes' rule [12]: $P(z = k|d) \propto P(z = k) \prod_{i=1}^{n} P(w_i|z = k)$.

10. http://www.csie.ntu.edu.tw/˜cjlin/liblinear/

7.2 Evaluation of Batch BTM

7.2.1 Topic Coherence

To evaluate the quality of topics discovered, we calculated the average PMI-Score, i.e., $\frac{1}{K}\sum_k \text{PMI-Score}(k)$, for each method. Table 4 lists the results on the Question and Tweets2011 collections with the number of most probable words T ranging from 5 to 20. We can see that BTM outperforms all the other methods consistently, and the improvement over LDA is significant (P-value < 0.01)[11]. Mix also produces better topics than LDA, but the improvement is less significant than BTM. LDA-U improves LDA moderately on the Tweets2011 collection, but surprisingly falls behind Mix. The results show that BTM can discover more coherent topics than the other three methods. Meanwhile, LDA cannot learn the topics very well from the short texts, and aggregating documents cannot fully resolve the sparsity problem.

We further investigated the content of the topics for qualitative analysis. Due to space limitation, we randomly drew two common topics existing in all the results of the methods using the strategy described in [41] for illustration. For each topic, we list its 20 most probable words, which are most representative for a topic. Besides, we also investigated 20 less probable words[12] in these topics to examine the coherence of a topic more comprehensively.

The two topics are about "job" and "snow", as listed in Tables 6 and 7. In the tables, the italic words are not directly relevant with "job" or "snow" judged by human. In Table 6, we can easily identify that these topics are about job from the top words for each method. However, in LDA, there are some words, such as "web", "website", and "google", more related to the website topic, rather than job. The results in LDA-U and Mix seem a little better than LDA, but still include a few of less relevant words such as "website" and "www". While in BTM, the 20 most probable words are more prominent and relevant about "job". For the less probable words, we find LDA includes the least words about "job". On the contrary, BTM includes more relevant words about "job" than the others, showing that the topic discovered by BTM is more coherent. The same phenomenon can be observed in Table 7. The above results indicate that the topics discovered by BTM are more prominent and coherent than the other methods over short texts.

7.2.2 Document Classification

Figure 3 shows the classification results on the Question and Tweets2011 collections. We find that 1) BTM always dominates the two baselines in the two collections significantly (P-value < 0.01)[13]. 2) Mix outperforms LDA on the Question collection, but falls behind LDA on the Tweets2011 collection. The possible reason is that the average document length of the Question collection is much shorter than the Tweets2011 collection, and thus a document is likely to involve one topic which fits the assumption of mixture of unigrams. 3) The improvement of LDA-U over LDA is not so much as shown in work [10]. This might result from the fact that in average there are less tweets posted per user in our dataset than theirs. Specifically, we find that about 63.3% of users posted only one tweet, while only 2.1% of users posted more than 9 tweets. Thus it is not strange that aggregating tweets by users has limited effect on topic learning.

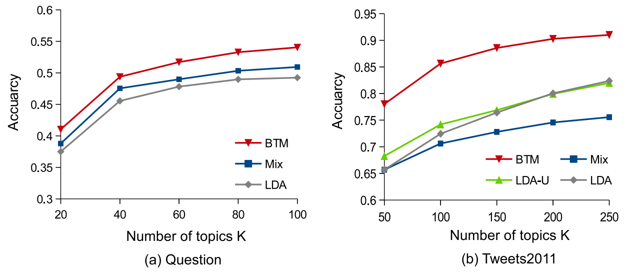


Fig. 3. Comparison of classification performance w.r.t. different numbers of topics on (a) the Question and (b) Tweets2011 collections.

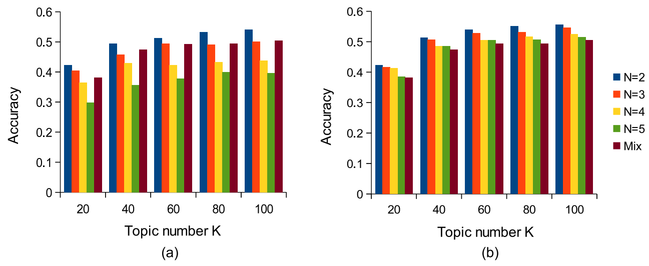


Fig. 4. Comparison of N-term topic model (NTM) trained over (a) the unweighted N-terms, (b) the weighted N-terms.

7.2.3 Biterm VS. N-term

BTM takes each biterm as a semantic unit that exhibits a single topic. A natural question is how about taking longer term combinations, such as tri-terms, as a semantic unit for topic modeling. To answer this question, we extend the biterm topic model to N-term topic models (NTM) by replacing biterm with N-term (i.e., an unordered group of N words co-occurring in the same document)[14]. In particular, when $N = 2$ it is exactly the BTM; When N is equal to or larger than the maximum length of documents in the collection,

11. We conducted two-sample T-test on the PMI-scores evaluated over different runs of LDA, denoted as S_{LDA}, and BTM, denoted as S_{BTM}, with null hypothesis H_0: $S_{LDA} < S_{BTM}$.

12. We simply select the words ranked from 1001 to 1020 in descending order of probability in the topic for illustration.

13. We conducted two-sample T-test on the accuracies evaluated over different runs of LDA, denoted as A_{LDA}, and BTM, denoted as A_{BTM}, with null hypothesis H_0: $A_{LDA} < A_{BTM}$.

14. Note here $N > 1$, since $N = 1$ means modeling all the words in the collection as independent.

TABLE 6
The 20 most probable words (second row) and 20 less probable words (third row) in topics about "job" from Tweets2011. The italic words are not directly relevant with "job" judged by human.

LDA	LDA-U	Mix	BTM
job jobs business web *website google design* online marketing *site blog* project manager search *www* company service sales services *post*	job jobs *design* manager project *web website site* business service company hiring *www* support sales services london *blog* senior engineer	jobs job business marketing *social media* online *web design website* manager *blog* project seo internet sales tips company *site* hiring	jobs job manager business sales hiring service services project company senior engineer management marketing nurse office assistant center customer development
nonprofit gallery announced presence published converting select reps requirement mgr *territory* recruiters *power involved announce poster larry dynamics feeds bristol*	expertise unemployed med iii *host* educational *fort tags apps assignments* labor *introduction* leads *github* assurance *avon manchester starting automotive table*	understand rep industrial *sustainability rankings* scholarships stay *single* campus extra *cheap* 101 vp *relationships* beginners colorado compliance *face winning* mechanical	springfield mlm recruit *oil* req unemployment processing *overview* awards recruiters *ict finish* entrepreneur comp assist 1000 *alliance* locations patent auditor

TABLE 7
The 20 most probable words (second row) and 20 less probable words (third row) in topics about "snow" from Tweets2011. The italic words are not directly relevant with "snow" judged by human.

LDA	LDA-U	Mix	BTM
snow *car* weather cold *drive* storm winter ice *road bus driving* rain *ride* traffic *cars safe* closed due warm *train*	snow weather cold winter ice storm rain stay warm *due car closed* coming spring *drive traffic safe* sun blizzard city	snow weather cold storm winter ice rain warm degrees stay sun spring safe blizzard coming wind cyclone *chicago* freezing *inches*	snow cold weather early stay ready ice winter storm hour hours *weekend* warm late coming spring rain *tired* sun hot
western *dmv* covering *a4* push pulling *milwaukee* remains *pace idiots* 95 *commuter buick owner cta transmission cyclist* flurries *camping tyre*	locations sunset drizzle *mississippi interstate* residents *portland students* fireplace *letting yuck ton counties* signal *counting* blankets pushed *3pm springfield venture*	*australian thankful* station *stops groundhogday* possibly *cleveland traveling sidewalk* covering predicting ten *grass meant double* affect *zoo schedule* blew *causing*	temperature cyclone warmth issued colder *mood couch* snows pre *traveling polar outages* umbrella filled *yawn outage* flurries online gloves speed

NTM is equivalent to Mix, since all the words in a document share the same topic.

Figure 4(a) compares the performance of NTM by document classification on the Question collection. We find that as N increases, the performance of NTM decreases. This is reasonable since the assumption that all the N words in a N-term belong to a single topic becomes too strict when N is large. However, it is strange that NTM even underperforms Mix when N is larger than 3. Further investigation found that the numbers of N-terms generated from documents with different lengths vary greatly. Note that a document with L words will generate $\binom{L}{N}$ N-terms, which grows dramatically as L increases. For example, when $N = 3$, a document with 3 words will generate one tri-term, while a document with 10 words will generate 120 tri-terms. As a consequence, N-terms generated from lengthy documents will dominate the training data, and thus hurt the performance.

A simple way to amend this problem is to add a weight $1/\binom{L-1}{N-1}$ for each N-term, where L is the length of its document. In this way, N-terms in longer documents will receive a smaller weight, and the word distribution of the corpus will stay unchanged. We re-ran NTM over the weighted N-terms[15], and show its result in Figure 4(b). We can see that the performance of NTM still decreases gradually as N grows, but now approaches to Mix. Compared with Figure 4(a), we can see that the performance of NTM is improved substantially by using the weighted N-terms when $N > 2$. However, the improvement is slight when $N = 2$, since the number of biterms generated from documents with different lengths do not vary much.

15. In the Gibbs sampling procedure, here we only need to update the counts of topic assignments using the weights of biterms, without changing the inference algorithm.

TABLE 8
Time cost (seconds) per iteration of BTM and LDA on Tweets2011 collection.

K	50	100	150	200	250
LDA	38.07	74.38	108.13	143.47	178.66
BTM	128.64	250.07	362.27	476.19	591.24

TABLE 9
Memory cost (megabytes) per iteration of BTM and LDA on Tweets2011 collection.

K	50	100	150	200	250
LDA	3177	5524	7890	10218	12561
BTM	927	946	964	984	1002

7.2.4 Efficiency Comparison

For efficiency comparison, we list the average running time (per iteration) of BTM and LDA in our

TABLE 10
PMI-Scores of the online algorithms. A larger PMI-Score indicates more coherent topics.

	K	50			100		
Collection	Method	Top5	Top10	Top20	Top5	Top10	Top20
Tweets2011	iLDA	2.53 ± 0.04	2.00 ± 0.05	1.59 ± 0.03	2.50 ± 0.04	1.97 ± 0.05	1.55 ± 0.02
	oBTM	2.63 ± 0.03	2.13 ± 0.03	1.80 ± 0.02	$\mathbf{2.72 \pm 0.03}$	2.16 ± 0.03	1.80 ± 0.02
	iBTM	$\mathbf{2.72 \pm 0.03}$	$\mathbf{2.17 \pm 0.02}$	$\mathbf{1.83 \pm 0.02}$	2.71 ± 0.03	$\mathbf{2.18 \pm 0.02}$	$\mathbf{1.83 \pm 0.04}$
Weibo	iLDA	2.37 ± 0.05	1.95 ± 0.02	1.69 ± 0.02	2.43 ± 0.05	1.90 ± 0.02	1.70 ± 0.03
	oBTM	$\mathbf{2.49 \pm 0.04}$	$\mathbf{2.02 \pm 0.02}$	$\mathbf{1.75 \pm 0.04}$	2.50 ± 0.03	$\mathbf{1.95 \pm 0.02}$	1.74 ± 0.04
	iBTM	2.48 ± 0.05	2.01 ± 0.02	1.74 ± 0.03	$\mathbf{2.54 \pm 0.04}$	$\mathbf{1.95 \pm 0.02}$	$\mathbf{1.75 \pm 0.05}$

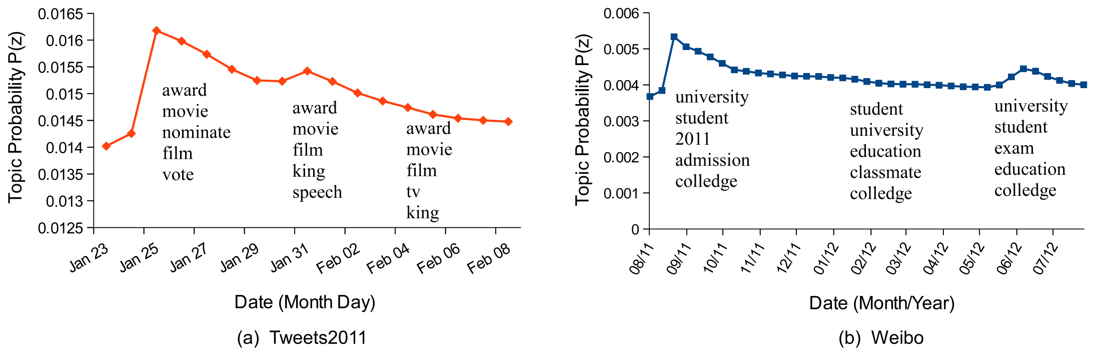


Fig. 5. Examples of topic evolution discovered by iBTM from the (a) Tweets2011 and (b) Weibo collections.

experiments on the collection Tweets2011 in Table 8. We can see that the running time of BTM is always about 3 times of LDA over different topic numbers. Table 9 shows the overall memory cost of BTM and LDA on the same collection. We observe that the memory required by LDA rapidly increases as the topic number K grows, which is more than 10 times of BTM when K is larger than 200. As opposed to LDA, the memory required by BTM grows very slowly. With further investigation, we find the major part of memory cost of BTM is to store the biterms, which is not sensitive to the topic number K. We also found similar results from the Question collection (Note that the results are not shown here due to space limitation).

7.3 Evaluation of Online Algorithms of BTM

7.3.1 Topic Coherence

To evaluate the topic quality of online algorithms, we compare the average PMI-Scores of the three models, i.e., oBTM, iBTM and iLDA, on Tweets2011 and Weibo collections. The results are shown in Table 10. We find that the PMI-Scores of oBTM and iBTM are very close, and both of them outperform iLDA consistently. Meanwhile, by comparing oBTM and iBTM, we found no dominant results between them.

We also find that the online algorithms of BTM can capture the topic evolution in text streams. Figure 5 (a) illustrates an example about the topic of 2011 Academy Awards (also called Oscars Awards) in Tweets2011 collection (K=50). Below the curve, we presented the 5 most probable words at different time points. We observe that this topic reaches its peek of interest on January 25, when the nominees were announced. In the next few days, the interest decreased until Jan 31, when the nominated movie "The King's Speech" won the Screen Actors Guild Awards. Figure 5 (b) provides another example from the Weibo collection (K=100)[16], and the topic is about college education. This topic became popular from August to September in 2011, when the college admission carried on. Additionally, this topic also emerged in June 2012 since the college entrance examination was taken place in that period.

7.3.2 Document Classification

We further compare the classification performance of the online algorithms on Tweets2011 (K=50) and Weibo (K=100) collections with respect to the number of days processed, depicted in Figure 6. In this Figure, "Batch BTM" denotes running a batch BTM over data received up to time t. Due to the expensive computational cost of the batch BTM, we only run the batch algorithm up to 80 days on the Weibo collection. From Figure 6, we have the following observations. Overall, the accuracy of oBTM and iBTM are close to the batch BTM, but consistently higher than the accuracy of iLDA. With more data received, oBTM and iBTM improve their performance substantially at the beginning, and then become stable gradually. In contrast, the accuracy of iLDA does not increase steadily, or even decreases in some cases. This phenomenon was also found in [16], yet it seems to be worse on short texts. Comparing iBTM with oBTM,

16. The 5 most probable words are translated from Chinese.

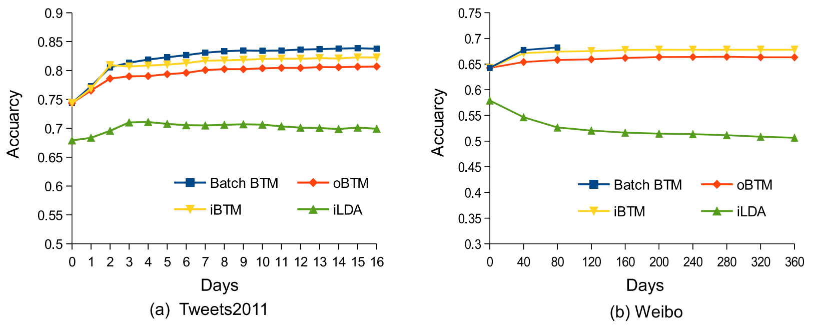


Fig. 6. Comparison of classification performance of online algorithms on the Tweets2011 and Weibo collections.

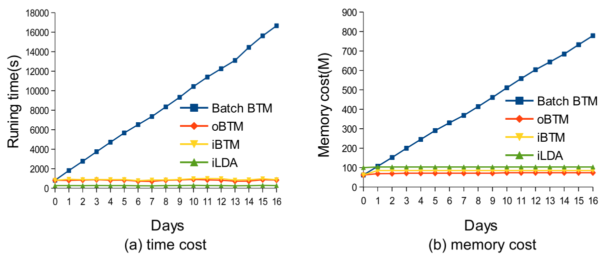


Fig. 7. Time and memory cost comparison on the Tweets2011 collection.

we find iBTM works slightly better than oBTM in this task, indicating that processing the data in sequential order might better fit the text stream data.

7.3.3 *Efficiency Comparison*

To demonstrate the efficiency of the online algorithms, we plot the time and memory cost of the batch BTM, iLDA, oBTM, and iBTM in Figure 7 on Tweets2011 dataset with $K = 50$. We can see that both the time and memory cost of the three online algorithms stay constant as more data received, while oBTM and iBTM cost less memory than iLDA. In contrast, the cost of batch BTM increases linearly as the time slice t grows. The results are consistent with the complexity analysis in Table 2. It demonstrates that the two online algorithms of BTM can be efficiently employed to learn topics from large datasets.

8 Conclusion

Topic modeling over short texts is an increasingly important task due to the prevalence of short texts on the Web. Compared with normal texts, short texts bring severe sparsity problems for conventional topic models. As the first attempt, we propose a novel topic model for general short texts, namely the biterm topic model (BTM). BTM can well capture the topics within short texts by explicitly modeling word co-occurrence patterns in the whole corpus. Besides, we also introduce two online algorithms for BTM, which are efficient to handle large scale datasets. Experimental results on real-word short text datasets show that BTM can learn higher quality topics, and better infer the documents' topic proportions than state-of-the-art methods. Besides, BTM is simple and easy to implement, and also scales up well via the proposed online algorithms. All these benefits make BTM a promising tool for content analysis on short texts for various applications, such as recommendation, event tracking, and text retrieval, etc.

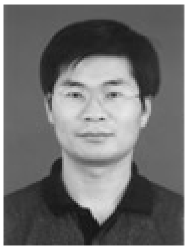
9 Acknowledgements

This work was funded by the 973 Program of China under Grants No. 2014CB340401 and No. 2012CB316303, 863 Program of China under Grants No. 2012AA011003, National Natural Science Foundation of China under Grant No. 61232010, No. 61173064, No. 61202213, No. 61203298, and National Key Technology R&D Program under Grant No. 2012BAH39B04.

References

[1] D. Ramage, S. Dumais, and D. Liebling, "Characterizing microblogs with topic models," in *International AAAI Conference on Weblogs and Social Media*, vol. 5, no. 4, 2010, pp. 130–137.
[2] W. Zhao, J. Jiang, J. Weng, J. He, E. Lim, H. Yan, and X. Li, "Comparing Twitter and traditional media using topic models," *Advances in Information Retrieval*, pp. 338–349, 2011.
[3] J. Guo, G. Xu, X. Cheng, and H. Li, "Named entity recognition in query," in *SIGIR*. ACM, 2009, pp. 267–274.
[4] J. Weng, E. Lim, J. Jiang, and Q. He, "Twitterrank: finding topic-sensitive influential twitterers," in *WSDM*. ACM, 2010, pp. 261–270.
[5] C. X. Lin, B. Zhao, Q. Mei, and J. Han, "PET: a statistical model for popular events tracking in social communities," in *SIGKDD*. ACM, 2010, pp. 929–938.
[6] T. Hofmann, "Probabilistic latent semantic indexing," in *SIGIR*. ACM, 1999, pp. 50–57.
[7] D. Blei, A. Ng, and M. Jordan, "Latent Dirichlet allocation," *JMLR*, vol. 3, pp. 993–1022, 2003.
[8] J. Boyd-Graber and D. M. Blei, "Syntactic topic models," Tech. Rep. arXiv:1002.4665, Feb 2010.
[9] X. Wang and A. McCallum, "Topics over time: a non-Markov continuous-time model of topical trends," in *SIGKDD*. NY, USA: ACM, 2006, pp. 424–433.
[10] L. Hong and B. Davison, "Empirical study of topic modeling in Twitter," in *Proceedings of the First Workshop on Social Media Analytics*. ACM, 2010, pp. 80–88.
[11] H. Lakkaraju, I. Bhattacharya, and C. Bhattacharyya, "Dynamic multi-relational Chinese restaurant process for analyzing influences on users in social media," in *ICDM*. Washington, DC, USA: IEEE Computer Society, 2012, pp. 389–398.
[12] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Machine learning*, vol. 39, no. 2, pp. 103–134, 2000.

[13] A. Gruber, M. Rosen-Zvi, and Y. Weiss, "Hidden topic Markov models," *AISTATS*, 2007.
[14] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the bayesian restoration of images," *PAMI*, no. 6, pp. 721–741, 1984.
[15] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, *An introduction to variational methods for graphical models*. Springer, 1998.
[16] K. R. Canini, L. Shi, and T. L. Griffiths, "Online inference of topics with latent Dirichlet allocation," in *AISTATS*, 2009, pp. 65–72.
[17] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *JASIST*, vol. 41, no. 6, pp. 391–407, 1990.
[18] D. Blei, "Probabilistic topic models," *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.
[19] H. M. Wallach, "Topic modeling: beyond bag-of-words," in *ICML*. ACM, 2006, pp. 977–984.
[20] D. Newman, E. V. Bonilla, and W. Buntine, "Improving topic coherence with regularized topic models," in *NIPS*, 2011, pp. 496–504.
[21] D. Mimno, H. Wallach, E. Talley, M. Leenders, and A. McCallum, "Optimizing semantic coherence in topic models," in *EMNLP*. ACL, 2011, pp. 262–272.
[22] X. Phan, L. Nguyen, and S. Horiguchi, "Learning to classify short and sparse text & web with hidden topics from large-scale data collections," in *WWW*. ACM, 2008, pp. 91–100.
[23] O. Jin, N. Liu, K. Zhao, Y. Yu, and Q. Yang, "Transferring topical knowledge from auxiliary long texts for short text clustering," in *CIKM*. ACM, 2011, pp. 775–784.
[24] O. Phelan, K. McCarthy, and B. Smyth, "Using Twitter to recommend real-time topical news," in *Proceedings of the third ACM conference on Recommender systems*. NY, USA: ACM, 2009, pp. 385–388.
[25] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi, "Short and tweet: experiments on recommending content from information streams," in *CHI*. ACM, 2010, pp. 1185–1194.
[26] Y. Wang, E. Agichtein, and M. Benzi, "TM-LDA: efficient online modeling of latent topic transitions in social media," in *SIGKDD*. NY, USA: ACM, 2012, pp. 123–131.
[27] X. Yan, J. Guo, S. Liu, X.-q. Cheng, and Y. Wang, "Clustering short text using ncut-weighted non-negative matrix factorization," in *CIKM*. NY, USA: ACM, 2012, pp. 2259–2262.
[28] X. Yan, J. Guo, S. Liu, X. Cheng, and Y. Wang, "Learning topics in short texts by non-negative matrix factorization on term correlation matrix," in *SDM*. SIAM, 2013.
[29] X. Yan, J. Guo, Y. Lan, and X. Cheng, "A biterm topic model for short texts," in *WWW*. ACM, 2013, pp. 1445–1456.
[30] T. Griffiths and M. Steyvers, "Finding scientific topics," *PNAS*, vol. 101, no. Suppl 1, pp. 5228–5235, 2004.
[31] W. Zhao, J. Jiang, J. He, Y. Song, P. Achananuparp, E. Lim, and X. Li, "Topical keyphrase extraction from Twitter," in *ACL*, 2011, pp. 379–388.
[32] J. S. Liu, "The collapsed gibbs sampler in Bayesian computations with applications to a gene regulation problem," *JASA*, vol. 89, no. 427, pp. 958–966, 1994.
[33] G. Heinrich, "Parameter estimation for text analysis," *Technical report*, 2005.
[34] L. AlSumait, D. Barbará, and C. Domeniconi, "On-line LDA: Adaptive topic models for mining text streams with applications to topic detection and tracking," in *ICDM*. IEEE, 2008, pp. 3–12.
[35] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. springer New York, 2006, vol. 1.
[36] B. Marthi, H. Pasula, S. Russell, and Y. Peres, "Decayed MCMC iltering," in *UAI*, 2002, pp. 319–326.
[37] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent Dirichlet allocation," in *NIPS*, 2010, pp. 856–864.
[38] T. Griffiths, M. Steyvers, D. Blei, and J. Tenenbaum, "Integrating topics and syntax," *NIPS*, vol. 17, pp. 537–544, 2005.
[39] J. Boyd-Graber, J. Chang, S. Gerrish, C. Wang, and D. Blei, "Reading tea leaves: How humans interpret topic models," in *NIPS*, 2009.
[40] D. Newman, J. H. Lau, K. Grieser, and T. Baldwin, "Automatic evaluation of topic coherence," in *NAACL*, 2010.
[41] D. Cai, Q. Mei, J. Han, and C. Zhai, "Modeling hidden topics on document manifold," in *CIKM*. ACM, 2008, pp. 911–920.



Xueqi Cheng is a Professor at the Institute of Computing Technology, Chinese Academy of Sciences (ICT-CAS), and the director of the Research Center of Web Data Science & Engineering (WDSE) in ICT-CAS. His main research interests include Network Science, Web Search and Data Mining, Big Data Processing and Distributed Computing Architecture, etc. He has published over 100 publications in prestigious journals and conferences, including IEEE ToIT, IEEE TKDE, Journal of Statistics Mechanics: Theory and Experiment, Physical Review E., ACM SIGIR, WWW, ACM CIKM, WSDM, IJCAI, ICDM and so on. He has won the Best Paper Award in CIKM (2011) and Best Student Paper Award in SIGIR (2012). He is currently serving on the editorial board of Journal of Computer Science and Technology, Journal of Computer, etc. XueQi Cheng is a recipient of the China Youth Science and Technology Award (2011), the Young Scientist Award of Chinese Academy of Sciences (2010), CVIC Software Engineering Award (2008), the Second prize for the National Science and Technology Progress (2004), etc.

Xiaohui Yan received the BSc degree in computer science and technology from the Central South University, Changsha, China, in 2008. He is currently working toward the Ph.D. degree in the Institute of Computing Technology at Chinese Academy of Science. His current research interests include text mining and machine learning.

Yanyan Lan is an associate researcher in Institute of Computing Technology, Chinese Academy of Sciences (ICT-CAS). Her research interests include learning to rank, statistical machine learning theory and data mining. She has published several papers on ICML, NIPS, WWW, SIGIR, CIKM and WSDM, etc. She is the coauthor of the best student paper for SIGIR 2012, and a program committee member of many international conferences or journals, such as SIGIR, KDD, AIRS and ACM TIST. Prior to joining ICT, she obtained her Ph. D. in probability and statistics from Academy of Mathematics and System Sciences, Chinese Academy of Sciences (AMSS-CAS). She won "Microsoft Research Asia Fellow" and "President's Scholarship of AMSS-CAS" in 2008.

Jiafeng Guo is an associate researcher in the Institute of Computing Technology, the Chinese Academy of Sciences (ICT-CAS). His major research interests include Web search and data mining, user data modeling, machine learning, and social search. He has published more than 50 papers in international journals and conferences, including IEEE TKDE, Physical Review E, Journal of Statistical Mechanics, SIGIR, WWW, CIKM, ICDM, WSDM, and IJCAI. He has won the Best Paper Award in CIKM (2011) and Best Student Paper Award in SIGIR (2012). He currently serves on the program committees of several international conferences and journals, including SIGIR, WWW, CIKM and TOIS. He was the recipient of the "Excellence Ph. D. Dissertation Award of CAS" in 2009.