A Novel Fast Framework for Topic Labeling Based on Similarity-preserved Hashing

Xian-Ling Mao[♠], Yi-Jing Hao[♠], Qiang Zhou[♠], Wen-Qing Yuan[♡], Liner Yang[♣], Heyan Huang[♠*]
[♠]Department of Computer Science, Beijing Institute of Technology, China
[♡]Beijing YanZhiYouWu Technology Co., LTD, China
[♣]Department of Computer Science and Technology, Tsinghua University, China
{maoxl, hyj, qzhou, hhy63}@bit.edu.cn
yuan@yzyw-inc.com, lineryang@gmail.com

Abstract

Recently, topic modeling has been widely applied in data mining due to its powerful ability. A common, major challenge in applying such topic models to other tasks is to accurately interpret the meaning of each topic. Topic labeling, as a major interpreting method, has attracted significant attention recently. However, most of previous works only focus on the effectiveness of topic labeling, and less attention has been paid to quickly creating good topic descriptors; Meanwhile, it's hard to assign labels for new emerging topics by using most of existing methods. To solve the problems above, in this paper, we propose a novel fast topic labeling framework that casts the labeling problem as a k-nearest neighbor (KNN) search problem in probability distributions. Our experimental results show that the proposed sequential interleaving method based on locality sensitive hashing (LSH) technology is efficient in boosting the comparison speed among probability distributions, and the proposed framework can generate meaningful labels to interpret topics, including new emerging topics.

1 Introduction

A wealth of topic models have been proposed to extract interesting topics in the form of multinomial distributions from the corpus automatically, which are useful data mining tools for the statistical analysis of document collections and other discrete data. A common, major challenge in applying all such topic models is to accurately interpret the meaning of each topic. In general, it is very difficult for users to understand a topic merely based on the multinomial word distribution, especially when they are not familiar with the background knowledge. Topic labeling, which generates meaningful labels for a topic so as to facilitate topic interpretation, has attracted increasing attention recently.

Early research on topic labeling generally either select top words in the distribution as primitive labels (Blei et al., 2003; Ramage et al., 2009), or generate labels manually in a subjective manner (Mei et al., 2006; Mei and Zhai, 2005). However, it is highly desirable to automatically generate meaningful labels. Several automatical labeling methods have been proposed recently (Mei et al., 2007; Lau et al., 2010; Lau et al., 2011; Magatti et al., 2009; Mao et al., 2012; Hulpus et al., 2013; Mehdad et al., 2013; Cano et al., 2014). These existing approaches generally take following steps to generate meaningful labels for a given topic: (1) extract candidate labels; (2) rank candidate labels. First, most of existing methods extract candidate labels from a document collection by natural language processing techniques for a given topic, which is time-consuming; for example, the candidate labels in the method proposed by Mei et al. (2007) are extracted from a reference collection using chunking and statistically important bigrams, which is time-consuming. Second, most of existing approaches depend on external knowledge sources such as Wikipedia and Google Directory etc, which cannot be used to label new emerging topics learned from a stream, such as Twitter, because there is no timely information about the new emerging topics in Wikipedia or Google Directory; for example, the method proposed by Lau et al. (2011) uses Wikipedia article titles as candidate labels, which is hard to assign correct labels for the new emerging topics in Twitter because there may not be the timely articles about the new emerging topics in Wikipedia. Thus, despite the success of these works, they are either time-consuming or hard to assign labels for new emerging topics.

*Corresponding author.

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 3339–3348, Osaka, Japan, December 11-17 2016.

Thus, it is highly desirable to rapidly generate meaningful labels for a topic word distribution while assign labels for new emerging topics as correctly as possible. However, to the best of our knowledge, no existing method has been proposed to satisfy the demand, except that the simplest method which uses top-n words in the distribution to interpret a topic. In this paper, we study this fundamental problem which most topic models suffer from, and propose a labeling framework to rapidly label a topic while assign labels for new emerging topics as correctly as possible.

Topics can be represented as vectors, i.e. j^{th} topic, T_j = (w_{1j}, w_{2j}, ..., w_{tj})^T. Each dimension corresponds to a separate word, and its value in the vector is the probability value. Due to the characters of distributions, we have two observations: (1) Similar topics all have higher probability values over the relevant words, thus a topic is near its similar topics in a probability vector set; For example, the topic "Military" and "Air Force" have both higher probability values over words like "soldier", "missile" and "death" than that over other words, except that the topic "Air Force" has higher probability values over words like "warcraft", "F22" and "pilot", than the topic "Military"; (2) Two different probability distributions might have the same label, because the inherent semantics of a distribution (i.e. topic) is not only reflected by the concrete probability values, but also by the focused words; for example, the following two topics don't have the same distributions, but they are the same topic, and the label is "Military".

	tank	bomb	...	death	take	cup	gun
$topic_1$	0.18	0.10	...	0.11	0.0002	0.007	0.08
$topic_2$	0.17	0.09	...	0.12	0.0003	0.009	0.07

Intuitively, when an event occurs, the information about the event exists in all kinds of sources (labeled data and unlabeled data), such as news, forums and social networks. Thus, these sources are parallel information channels. It is believed that the labels in labeled data will cover most topics in unlabeled data. Through the idea of parallel information channels, in a domain, if there is a database where each record consists of a topic word distribution and its corresponding label [1], and the database is updated uninterruptedly by learning from the latest labeled data as soon as possible; we can interpret a given topic by the labels of k-nearest distributions in the database (i.e. probability vector set) while assign labels for new emerging topics as correctly as possible. If the database is large, timely, and cover most kinds of topics in a domain, it's a reasonable labeling solution to label topics in the domain.

There are two challenges to achieve the intuition: (1) how to quickly find the k-nearest neighbours in a large probability vector set, given a high-dimensional topic word distribution; (2) how to update the database from the latest data as soon as possible.

We proposed a fast labeling framework to solve the two challenges: (1) to finding k-nearest distributions quickly, we use locality sensitive hashing (LSH) as a dimensionality reduction technique to accelerate distribution similarity comparison; (2) to accelerate the update speed of the database, we modify the batch learning algorithm for Labeled LDA to obtain an online learning algorithm.

2 Related work

In most existing research effects on statistical topic modeling, people generally either select top words in the distribution as primitive labels (Blei et al., 2003; Ramage et al., 2009; Ramage et al., 2011), or generate more meaningful labels manually in a subjective manner (Mei et al., 2006; Mei and Zhai, 2005). However, extracting top terms is not very useful to interpret the coherent meaning of a topic (Mei et al., 2007). Meanwhile manually generated labels require lots of human effort to generate such labels, and can easily be biased towards the user's subjective opinions.

Thus, several automatical labeling methods have been proposed recently. In 2007, Mei et al. first proposed probabilistic approaches to automatically label multinomial topics by extracting a set of candidate labels from a reference collection using chunking and statistically important bigrams, and the top ranked labels were chosen to represent the topic (Mei et al., 2007). Magatti et al. (2009) introduced an algorithm to label topics automatically according to a given category hierarchy. The hierarchy was obtained from the Google Directory and the OpenOffice English Thesaurus. The most similar label is assigned to the topic. Lau et al. (2010) proposed to label a topic via selecting one of the top-10 topic terms to label the

[1] Different distributions can have same label

overall topic by a reranking model. Different with their previous method (Lau et al., 2010), Lau et al. (2011) enlarged the candidate labels by making use of Wikipedia article titles.

Mao et al. (2012) proposed two effective algorithms that automatically assign concise labels to each topic in a hierarchy by exploiting sibling and parent-child relations among topics. The structured data in DBpedia is used to label topics (Hulpus et al., 2013). Mehdad et al. (2013) introduced a topic labeling approach that assigns the most representative phrases for a given set of sentences covering the same topic. Cano et al. (2014) proposed a summarisation framework to label the topics learned from Twitter, which is independent of external sources and only relies on the identification of dominant terms in documents related to the latent topic. Word embedding is also used to help label topics (Jin et al., 2016). Interestingly, Aletras and Stevenson (2013) proposed to label topics with images rather than text. Candidate images for each topic are retrieved from the web by querying a search engine using the top-n terms. The most suitable image is selected by using a graph-based algorithm.

Overall, these methods first extract the candidate labels, then rank these labels according to corresponding scoring function. The labeling framework is showed in Figure 1 (a). Despite the success of these works, they are either time-consuming or hard to assign labels for new emerging topics. Thus, in this paper, we will focus on the problems above, and proposed our fast topic labeling framework.

Figure 1: (a) Traditional topic labeling framework and (b) Fast topic labeling framework.

3 Fast Topic Labeling Framework

In this paper, we will study topic labeling problem from a different perspective, i.e. cast it as k-nearest neighbor (KNN) search problem in a probability vector set. As shown in Figure 1 (b), our framework consists of two main components described in the following sections.

3.1 Online Labeled LDA

One important component of our proposed framework is to construct a "topic-label" database which consists of probability distributions over words and corresponding labels, denoted as $DB_{m,n}$, m is the number of topics, and n is the size of vocabulary. We can use Labeled LDA (LLDA) proposed by Ramage et al. (2009) to construct the "topic-label" database. LLDA models the documents with labels, and obtains a probability distribution for each label. The training algorithm for LLDA runs in batch mode, and cannot update the database timely. However, in order that the framework can assign labels for new emerging topics as correctly as possible, the proposed framework needs to process latest labeled data timely, and add gradually new learned records into the "topic-label" database. Thus, we propose a online algorithm for Labeled LDA, called OLLDA, to accelerate the update speed of "topic-label" database.

We define the vector of corresponding labels of document d to be $\boldsymbol{\psi}^{(d)}$, and the number of labels to be M_d, i.e. $M_d = |\boldsymbol{\psi}^{(d)}|$. Similar to the batch variational inference for LDA (Blei et al., 2003), the batch variational inference for Labeled LDA approximates the true posterior by a simpler distribution $q(z, \theta, \beta)$, which is indexed by a set of free parameters. These parameters are optimized by maximizing the lower bound:

$$\log p(\boldsymbol{w}|\alpha, \eta, \Psi) \geq L(\boldsymbol{w}, \phi, \gamma, \boldsymbol{\lambda}, \Psi) \triangleq \mathbb{E}_q[\log p(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\beta}, \Psi|\alpha, \eta)] - \mathbb{E}_q[\log q(\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\beta})]. \quad (1)$$

To maximize the lower bound, we have to minimize the KL divergence between $q(\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\beta})$ and the

posterior $p(\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\beta}, \Psi|\boldsymbol{w}, \alpha, \eta)$. The distribution $q(\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\beta})$ can be fully factorized into the form:

$$q(z_{di} = l) = \phi_{dw_{di}l}; \qquad q(\theta_d) = \text{Dir}(\theta_d; \gamma_d); \qquad q(\beta_l) = \text{Dir}(\beta_l; \gamma_l), \tag{2}$$

where l is the index of a label, and also the index of corresponding topic. The posterior over the per-word topic assignments z is parameterized by ϕ, the posterior over the per-document topic weights θ is parameterized by $\boldsymbol{\gamma}$, and the posterior over the topics β is parameterized by $\boldsymbol{\lambda}$. Equation (1) factorizes to

$$\begin{aligned}
L(\boldsymbol{w}, \boldsymbol{\phi}, \boldsymbol{\gamma}, \boldsymbol{\lambda}, \Psi) = & \sum_{d, z_d \in \Psi} \{\mathbb{E}_q[\log p(w_d|\theta_d, z_d, \beta)] + \mathbb{E}_q[\log p(z_d|\theta_d)] - \mathbb{E}_q[\log q(z_d)] + \mathbb{E}_q[\log p(\theta_d|\alpha)] \\
& - \mathbb{E}_q[\log q(\theta_d)] + (\mathbb{E}_q[\log p(\beta|\eta)] - \mathbb{E}_q[\log q(\beta)])/D\} \\
= & \sum_d \sum_w n_{dw} \sum_{l \in \boldsymbol{\psi}^{(d)}} \phi_{dwl}(\mathbb{E}_q[\log \theta_{dl}] + \mathbb{E}_q[\log \beta_{lw}] - \log \phi_{dwl}) - \log \Gamma(\sum_{l \in \boldsymbol{\psi}^{(d)}} \gamma_{dl}) + \sum_{l \in \boldsymbol{\psi}^{(d)}} \\
& (\alpha - \gamma_{dl})\mathbb{E}_q[\log \theta_{dl}] + \log \Gamma(\gamma_{dl}) + (\sum_{l \in \boldsymbol{\psi}^{(d)}} - \log \Gamma \sum_w \lambda_{lw}) + \sum_w (\eta - \lambda_{lw})\mathbb{E}_q[\log \beta_{lw}] \\
& + \log \Gamma(\lambda_{lw}))/D + \log \Gamma(M_d \alpha) - M_d \log \Gamma(\alpha) + (\log \Gamma(W\eta) - W \log \Gamma(\eta))/D \\
\triangleq & \sum_d l(n_d, \phi_d, \gamma_d, \boldsymbol{\lambda}, \psi^{(d)}),
\end{aligned} \tag{3}$$

where W is the size of the vocabulary and D is the number of documents. $l(n_d, \phi_d, \gamma_d, \boldsymbol{\lambda}, \psi^{(d)})$ denotes the contribution of document d to the lower bound of Formula (1). This reveals that the variational objective relies only on n_{dw}, the number of times word w appears in document d. Thus, an online inference algorithm for Labeled LDA can be derived.

We can use coordinate ascent to optimize L over the variational parameters $\boldsymbol{\phi}$, $\boldsymbol{\gamma}$, $\boldsymbol{\lambda}$:

$$\phi_{dwl} \propto exp\{\mathbb{E}_q[\log \theta_{dl}] + \mathbb{E}_q[\log \beta_{lw}]\}; \qquad \gamma_{dl} = \alpha + \sum_w n_{dw}\phi_{dwl}; \qquad \lambda_{lw} = \eta + \sum_d n_{dw}\phi_{dwl}. \tag{4}$$

As the d^{th} vector of word counts n_d is observed, we perform an EM algorithm to obtain optimal parameter values. Similar to the work (Hoffman et al., 2010), we use the weight $\rho_d \triangleq (\tau_0 + d)^{-\kappa}$ to control the rate at which old values of $\boldsymbol{\lambda}$ are forgotten and $\tau_0 \geq 0$ slows down the early iterations of the algorithm, where $\kappa \in (0.5, 1]$ is needed to guarantee convergence. The proposed online variational inference for Labeled LDA (OLLDA) is described in Algorithm 1.

Algorithm 1 Online Variational Inference for Labeled LDA

```
1: ρ_d = (τ_0 + d)^{-κ}
2: Initialize λ randomly.
3: for d = 0 to ∞ do
4:     E step:
5:     Initialize γ_dl = 1.
6:     repeat
7:         for each word w in document d do
8:             for each label l of document d do
9:                 Set φ_dwl ∝ exp{E_q[log θ_dl] + E_q[log β_lw]}
10:                Set γ_dl = α + Σ_w φ_dwl n_dw
11:            end for
12:        end for
13:    until (1/M_d) Σ_l |change in γ_dl| < 0.00001
14:    M step:
15:    Compute λ̃_lw = η + D n_dw φ_dwl
16:    Set λ = (1 − ρ_d)λ + ρ_d λ̃.
17: end for
```

3.2 Distribution Similarity Ranking

After constructing the "topic-label" database, we have to rank distributions for a given topic. To compute similarity between two distributions, there are many metrics, such as Kullback-Leibler divergence

(KL divergence, KL) and Jensen-Shannon Divergence (JSD). Generally speaking, the vocabulary of a distribution is large (over 100K) while the number of records in database is also large, thus it's costly in computing and storage. For example, assume that the number of vocabulary is 100,000, the number of topics is 100,000, and the size of a float is 4 bytes, we need at least 40G space to store distributions.

Locality Sensitive Hashing (LSH), as a dimensionality reduction technique, has been widely applied in large-scale data mining, such as near-duplicate webpage detection (Manku et al., 2007) and image retrieval (Vogel and Schiele, 2007). The key idea of LSH is to assign a number to each point in a metric space by a function h uniformly selected from a family of hashing functions H so that the probability of collision (i.e., assigned by an identical number) is much higher for points close to each other than those far apart (Charikar, 2002). LSH functions involve the creation of short signatures (fingerprints) for each vector (point) in space such that those vectors that are closer to each other are more likely to have similar fingerprints. LSH functions are generally based on randomized algorithms and are probabilistic.

Based on LSH technology, the proposed distribution similarity ranking method will take the following three steps, described below.

3.2.1 Initial Ranking

To accelerate distribution similarity comparison and decrease the storage space, we will choose two representative types of LSH, cosine hash family (**Simhash** (Charikar, 2002)) and the Euclidean hash family (**P-stable LSH** (Datar et al., 2004)), as initial distribution similarity metrics.

Given a probability distribution (i.e. a topic), after obtained the ranked list of topics by a distribution similarity metric (such as Simhash, P-stable LSH), the corresponding labels of returned topics can be used to label the given topic. Because it's useless to assign too many labels to a topic, our systems will return top-20 topics for each given topic.

3.2.2 Re-ranking

A distribution similarity metric does not consider the inherent property of topics. For example, two different points in a probability vector set might have the same label. Thus, we have to consider the nature of topics. Because top-n words of similar topics should be similar, so we will re-rank the order of topics by making use of overlap similarity between top-n word set of the given topic and each in top-20 returned topics. Top-n, as a parameter, will be selected by experiments.

3.2.3 Sequential Interleaving of Two Lists

Given two ranking lists generated by different methods, to obtain benefit from the both results, we consider an interleaving process on these two ranking lists to obtain a better ranking list. Lots of works have been done to interleave two ranking lists of documents in information retrieval area (Hofmann et al., 2011; Hofmann et al., 2012; Chukllin et al., 2015). We borrow the interleaving idea in information retrieval area to handle this problem, and propose a a novel interleaving algorithm, called Sequential Interleaving, to obtain a better ranking list by merging two ranking lists.

In a retrieved list generated by an algorithm, the ranking position of a topic in the list can be treated as a reflection of "confidence level", which is how "good" the algorithm thinks the topic is similar to the given topic. Intuitively, the confidence level $\mathrm{CL}(p, L_r)$ for the probability distribution p of the position r in the ranking list L, should be an inverse function form of the ranking position r. To define the function form of $\mathrm{CL}(p, L_r)$, we inspect the metrics used to evaluating a retrieved list in information retrieval. DCG is a ranking-aware metric which can effectively evaluate how relevant a ranking list is for the query. Its widely-used binary value form (Chapelle et al., 2012) is defined as:

$$\mathrm{DCG}(L) = \sum_{r=1}^{len(L)} \frac{rel_{L_r}}{\log_2(r+1)} \tag{5}$$

where rel_{L_r} is the relevance of the document ranked at r in the ranking list L. In this formula, DCG reflects total confidence level of all documents in the ranking list, thus we can define $\mathrm{CL}(p, L_r)$ as:

$$\mathrm{CL}(p, L_r) = \begin{cases} \frac{1}{\log_2(r+1)} & p \in L, \\ 0 & p \notin L. \end{cases} \tag{6}$$

where p is a probability distribution, and r is the ranking postion of p in the ranking list L.

Given a topic, there are two ranking lists of top-k similar probability distributions (i.e. topics), and the goal is to combine the two ranking lists to obtain a better top-k ranking list. We first obtain the union of the two ranking lists L_1 and L_2, then for each probability distribution p in the union, to compute the total confidence level of p by the following simple formula:

$$\text{TotalCL}(p, L_1, L_2) = \alpha \sum_{L_{1_i}=p} \text{CL}(p, L_{1_i}) + (1-\alpha) \sum_{L_{2_j}=p} \text{CL}(p, L_{2_j}) \tag{7}$$

where $L_{1_i} = p$ denotes the probability distribution p is at the position i in the ranking list L_1, and α is a prior weighting factor. Finally, we sort all the probability distributions by using $\text{TotalCL}(p, L_1, L_2)$, and then give the top-k similar interleaved probability distributions as final ranking result; and thus the labels of these distributions are the top-k candidate labels for the given topic. The proposed algorithm is called Sequential Interleaving. Because the proposed framework requires the high speed of k-nearest neighbor search for a given topic, only locality sensitive hashing algorithms satisfy the condition, thus we combine the re-ranking lists of Simhash and P-stable by proposed Sequential Interleaving method to obtain a ranking list, and abbreviated as si. In this paper, we fairly treat the re-ranking results of Simhash and P-stable by setting α = 0.5.

4 Experiments and results

In this section, we present the results of the efficiency and effectiveness of the proposed method over three data sets.

4.1 Experiment Setup

Data Sets: We explore three different genres of data sets: the Simulated data (**SIMU**), the Conference proceedings (**CONF**), the Twitter dataset (**TW**). To construct the first dataset, we simulated 10,000 distributions over 10,000 words ($DB_{10000,10000}$), and 100,000 distributions over 10,000 words ($DB_{100000,10000}$). After collecting 2,924 fullpapers of four conference (SIGIR, SIGKDD, CIKM, and WWW) proceedings from the year 2010 to 2013, from Google Scholar, conference u-disks and authors' homepage, we obtained the second dataset. The last dataset contains 2.1G microblogs with 3503 hashtags, which removed microblogs without hashtag and hashtags whose idf is less than 50, downloaded from Twitter website in six days. We built "topic-label" database over the last two data sets by our OLLDA algorithm. Specifically, for **CONF**, we trained OLLDA over the data from CIKM2013 to obtain probability distributions with labels as test queries, and trained OLLDA over the remaining data as "topic-label" database; for **TW**, to increase the number of points with same labels in a probability vector set, we split the dataset into 12 pieces according to the time order, and trained separately OLLDA over the $1^{th} \sim 11^{th}$ pieces as "topic-label" database, and over the 12^{th} piece as test queries. All test queries in two datasets can be as the new emerging topics. After training, the statistics of data sets are shown in Table 1 [2].

Baselines: Because proposed framework is totally different with existing methods, e.g., the input of most of existing methods is a topic and a collection which generates the topic, and the input of the proposed framework is only a topic, thus, we cannot compare them directly. Meanwhile there is no related work that focus on efficiency of topic labeling. Essentially, the core of the proposed framework is the comparison among distributions, thus, we choose KL divergence (KL) and Jensen-Shannon Divergence (JSD) as distribution similarity metrics in our framework, as baselines. All methods in this paper are denoted as $FR_{metric,ranking\_stage}$, where *metric* means which distribution similarity metric to choose, and *ranking_stage* means *Initial Ranking* or *Re-ranking*. We use "1" to denote "Initial Ranking" and "2" denote "Re-ranking". For example, assume that the distribution similarity metric is JSD, and just use initial ranking, the method can be denoted as $FR_{jsd,1}$. The sequential interleaving result of the ranking lists of $FR_{sh,2}$ and $FR_{ps,2}$ is denoted as FR_{si}.

All experiments were conducted on a server with dual 6-core Intel i7 cpus with 3.4Ghz, 32G memory.

[2] All "topic-label" databases have been published at "https://github.com/TopicLabeling/tldb".

4.2 Efficiency of Distribution Similarity Metrics over Simulated Data

We sample respectively 20 probability distributions from $DB_{10000,10000}$ and $DB_{100000,10000}$ as test queries and rank the distributions in corresponding SIMU dataset, then compute avarage time per query and space cost. Because the main cost in our distribution similarity ranking is spent in "Initial Ranking" stage, and the cost of re-ranking is little, thus only report the results in "Initial Ranking" stage. Table 2 shows a detailed comparison of $FR_{kl,1}$, $FR_{jsd,1}$, $FR_{sh,1}$ and $FR_{ps,1}$ on two SIMU datasets. We can make two observations from the table: the computing time and storage space of $FR_{ps,1}$ and $FR_{sh,1}$, are significantly less than that of baseline methods. In a word, the efficiency of LSH significantly outperforms the baselines in terms of time and space.

Table 1: The statistics of CONF and TW data sets.

	CONF	TW
Num. of Vocabulary	25,160	189,841
Num. of Labels	586	3,503
Num. of Distributions	957	12,139

Table 2: The efficiency of four methods over the SIMU datasets.

	$\mathbf{DB_{10000,10000}}$		$\mathbf{DB_{100000,10000}}$	
Methods	**Time (s)**	**Space (M)**	**Time (s)**	**Space (M)**
$FR_{kl,1}$	5.761	1,230.830	546.514	12,0206.272
$FR_{jsd,1}$	5.051	1,206.972	656.120	12,211.392
$FR_{sh,1}$	0.078	**115.996**	**0.504**	**674.750**
$FR_{ps,1}$	**0.075**	411.028	0.689	4,020.030

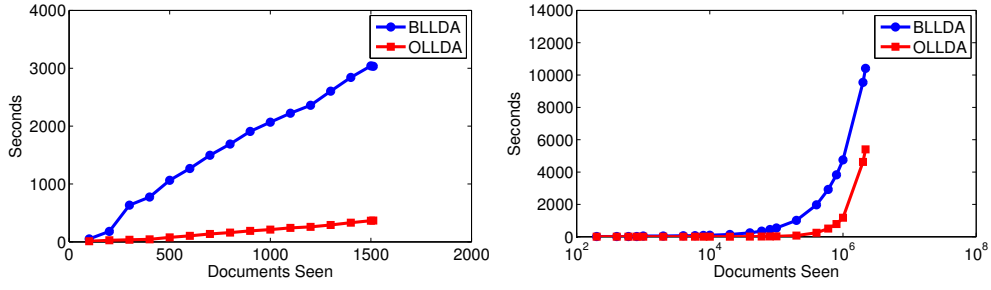
4.3 Performance of Proposed Framework over Real-world Data

As described above, the distributions in our "topic-label" database has their corresponding labels, which are used as standard labels (ground truth). We first show some sample results of our fast labeling methods in Table 4. For comparison, we also show the baselines' labels and standard labels for the same topics. It is clear that the rapidly generated labels can all capture the meaning of the topic to some extent; indeed, most of them are as good as standard labels (e.g., "social network" and "michael jackson"), though some are not (e.g., "text classification") but they are similar to standard labels.

4.3.1 Efficiency of OLLDA

OLLDA has several learning parameters: $\kappa \in (0.5, 1]$ and $\tau_0 \geq 0$. Similar to Hoffman's parameter choosing method (Hoffman et al., 2010), we also set $\kappa = 0.5$ and $\tau_0 = 64$ as the best learning parameter settings for both corpora.

In this experiment, we evaluated the efficiency of OLLDA, compared with batch LLDA (BLLDA). We simulated the situation that documents are coming in a stream. For our OLLDA, the model parameters are updated every time a new document arrives. We computed total training time cost at some points of the seen documents using OLLDA. As for BLLDA, we computed the training time cost at the same points, and note that each run has to use all of the documents previously seen.



(a) Conf (b) Twitter

Figure 2: Training Time On Two Real Datasets

When a new document arrives, BLLDA has to run over all observed documents for many iterations again. Since the time for each iteration grows with the number of documents, the total time for BLLDA grows fast. However, OLLDA only processes the new coming document and update parameters. Thus, as Figure 2 shows, BLLDA costs much more time to get the new parameters compared with OLLDA,

especially when the number of observed documents is large. When running over the entire Conf corpus, the training time of OLLDA is less than 400 seconds, while BLLDA takes more than 3,000 seconds, which is about 8 times longer. Over TW dataset, BLLDA takes more than 10,000 seconds, while OLLDA takes less than 6,000 seconds.

In Figure 2 (b), when the number of observed documents is large, the time cost increases quickly for OLLDA and BLLDA, because of the greater number of parameters and IO cost. However, OLLDA still performs better than BLLDA. If we use mini-batch trick and other optimize technologies in OLLDA, the efficiency of OLLDA will get greater improvement.

4.3.2 Effectiveness of Distribution Similarity Ranking

We compute the following performance metrics over CONF and TW dataset: (1) *Match at top N results (**Match@N**)*, which indicates whether the top N results contain any correct labels; (2) *Precision at top N results (**P@N**)*.

Table 3: The effectiveness of four methods over the CONF and TW datasets.

Datasets	Methods	Match@N				P@N			
		N=1	N=3	N=5	N=10	N=1	N=3	N=5	N=10
CONF	$FR_{kl,1}$	0.0000	0.0000	0.0069	0.0069	0.0000	0.0000	0.0014	0.0007
	$FR_{jsd,1}$	**0.1458**	**0.2569**	**0.2847**	**0.3819**	**0.1458**	**0.0949**	**0.0667**	**0.0458**
	$FR_{sh,1}$	0.0347	0.0556	0.0764	0.1667	0.0347	0.0208	0.0181	0.0181
	$FR_{sh,2}$	0.0833	0.1458	0.1597	0.1806	0.0833	0.0556	0.0375	0.0236
	$FR_{ps,1}$	0.0000	0.0208	0.0278	0.0556	0.0000	0.0069	0.0056	0.0056
	$FR_{ps,2}$	0.0556	0.0694	0.0903	0.1042	0.0556	0.0255	0.0208	0.0125
	FR_{si}	0.0972	0.1528	0.1736	0.2153	0.0972	0.0556	0.0389	0.0271
TW	$FR_{kl,1}$	0.0500	0.0500	0.1000	0.1000	0.0500	0.0500	0.0500	0.0350
	$FR_{jsd,1}$	0.9000	0.9000	**0.9500**	**1.0000**	0.9000	0.6000	**0.5300**	0.4550
	$FR_{sh,1}$	0.9000	**0.9500**	**0.9500**	0.9500	0.9000	**0.6167**	**0.5300**	**0.4600**
	$FR_{sh,2}$	0.9000	**0.9500**	**0.9500**	0.9500	0.9000	**0.6167**	**0.5300**	0.4500
	$FR_{ps,1}$	0.8500	0.9000	0.9000	0.9000	0.8500	0.6000	0.5200	0.4550
	$FR_{ps,2}$	0.9000	0.9000	0.9000	0.9000	0.9000	0.5833	0.5200	0.4500
	FR_{si}	**0.9500**	**0.9500**	**0.9500**	**1.0000**	**0.9500**	**0.6167**	**0.5300**	**0.4600**

Table 4: Sample topics and algorithm-generated labels from TW dataset.

	CONF			TW		
Stand. Label	**recommender_systems**	**social_network**	**text_classification**	**michaeljackson**	**treat**	**rugby**
$\mathbf{FR_{kl,1}}$ **(Top-3)**	personal_information_management recommender location_selection	location_selection sensemaking spreadsheets	personal_information_management image_clustering recommender	glamourkills twitdraw 140kingofpop	voss bonjovi tinychat	ff brazilmissesdemi dietalk
$\mathbf{FR_{jsd,1}}$ **(Top-3)**	collaborative_filtering matrix_factorization collaborative_filtering	social_networks social_networks privacy	query_classification graph_regularization active_learning	michaeljackson michaeljackson thisisit	trick fb happyhalloween	fb nhl rugby
$\mathbf{FR_{sh,1}}$ **(Top-3)**	collaborative_filtering matrix_factorization recommender_systems	social_networks social_networks social_search	query_classification domain_adaptation hierarchical_classification	michaeljackson michaeljackson uknowurblackwhen	trick story09 yeg	rugby rugby rugby
$\mathbf{FR_{ps,1}}$ **(Top-3)**	diversity personalization evaluation	topic_model social-network_analysis social_network	semi-supervised_learning machine_learning evaluation	michaeljackson mj fb	trick fb love	fb rugby fb
$\mathbf{FR_{si}}$ **(Top-3)**	collaborative_filtering matrix_factorization personalization	social_networks social-network_analysis social_networks	query_classification semi-supervised_learning hierarchical_classification	michaeljackson michaeljackson michaeljackson	trick trick treat	rugby rugby rugby
Top-10	users item rating items ratings recommendation collaborative filtering methods recommender	social network graph users content group nodes labels such people	training documents domain classification articles method text test used terms	michaeljackson jackson michael shirts ringtone jacko ringtones movie kingofpop our	brother's wapo today incredible trife strokes things when somebody sense	rugby davidarchie uia game beginnings try great after fatal be4

From Table 3, we can make several observations: (1) For both datasets, $FR_{x,2}$ performs better than $FR_{x,1}$, *x* denotes *sh* or *ps*, which shows that the step "Re-ranking" is effective. (2) For both datasets, FR_{si} performs better than $FR_{sh,2}$ and $FR_{ps,2}$, which shows that the step "Sequential Interleaving" is effective. (3) For both datasets, $FR_{jsd,1}$ performs better than $FR_{sh,1}$ and $FR_{ps,1}$ in most cases, which shows that JSD is good distribution similarity comparision method. However, the time cost of $FR_{jsd,1}$ is higher than LSH methods, showed in Table 5. (4) For both datasets, the performance of $FR_{ps,1}$ is alway poorer than that of $FR_{sh,1}$. Given a probability distribution $\vec{p}$, assume that $\mathbf{P}$ is the set of all probability

distributions, and the set $S_1 = \{p_i | \cos^{-1} \frac{\vec{p}_i \cdot \vec{p}}{\|\vec{p}_i\| \|\vec{p}\|} \leq \theta, p_i \in \mathbf{P}\}$, and assume that $\vec{p}_0$ is a probability distribution, which satisfies $p_0 = argmin_{p_i \in S_1} \|\vec{p}_i - \vec{p}\|$, and the set $S_2 = \{p_i | \ \|\vec{p}_i - \vec{p}\| \leq \ \|\vec{p}_0 - \vec{p}\|\}$. It's easy to prove that $S_2 \subseteq S_1$. That means: the point p_0 locates in the bound of S_1 and S_2, however, the set for Euclidean-based methods is subset of the one for angle-based methods, which means that the ability of finding KNN points for angle-based methods are better than the one for Euclidean-based methods in probability distributions. Thus, $FR_{sh,1}$ always performs better than $FR_{ps,1}$. (5) For CONF dataset, the metric values of all methods are not high. Because our metrics only count the labels which are same as standard labels, and ignore the similar labels. Furthermore, the ratio ($\frac{Num.of Distributions}{Num.of Labels}$) in CONF is small (showed in Table 1). Small ratio means that a lable has fewer points in a probability vector set, thus it's natural that the values of all metrics are very low. Thus, we evaluate all ranking results judged manually by three students, and take use of voting method to obtain the final results, showed in Table 6. The table shows that the metric values increase, and FR_{si} performs best. (6) For TW dataset, the performance of FR_{si} is the best among all methods. The improvements are significant by t-test at the 95% significance level. Thus, the overall results show that proposed distribution similarity ranking is effective. Meanwhile, because the test queries in two datasets can be as the new emerging topics, thus the results also show that the proposed method can handle the new emerging topics.

Table 5: The efficiency over the CONF and TW datasets.

	CONF	TW
Methods	**Time (s)**	**Time (s)**
$FR_{kl,1}$	0.0727	1.9823
$FR_{jsd,1}$	0.2310	10.0959
$FR_{sh,1}$	**0.0024**	**0.0541**
$FR_{ps,1}$	0.0260	0.2594

Table 6: The results of human judge over CONF dataset.

Methods	**Match@N**				**P@N**			
	N=1	**N=3**	**N=5**	**N=10**	**N=1**	**N=3**	**N=5**	**N=10**
$FR_{kl,1}$	0.100	0.200	0.200	0.500	0.100	0.067	0.050	0.065
$FR_{jsd,1}$	0.350	0.650	0.700	0.900	0.350	0.467	0.360	0.295
$FR_{sh,1}$	0.350	0.600	0.700	0.750	0.350	0.300	0.280	0.210
$FR_{ps,1}$	0.200	0.250	0.450	0.800	0.200	0.167	0.150	0.175
FR_{si}	**0.467**	**0.750**	**0.831**	**0.950**	**0.467**	**0.530**	**0.455**	**0.375**

4.3.3 Efficiency of Distribution Similarity Ranking

For each query in test set, we rank the distributions in corresponding dataset, then compute avarage time per query. Table 5 shows a detailed comparison of $FR_{sh,1}$, $FR_{ps,1}$, $FR_{kl,1}$ and $FR_{jsd,1}$ on **TW** and **CONF**. We can make following observations: the computing time of $FR_{ps,1}$ and $FR_{sh,1}$, are significantly less than other methods. In a word, the efficiency of the proposed distribution similarity ranking method based on LSH significantly outperforms the baselines in terms of time.

5 Conclusion

In this paper, we cast the rapid labeling problem as a k-nearest neighbor (KNN) search problem among existing "topic-label" database, which is updated uninterruptedly by online learning from the latest data. The experimental results show that the proposed framework can generate meaningful labels that are useful for interpreting topics in real time.

Acknowledgements

The work was supported by 863 Program of China (No. 2015AA015404) and NSFC (No. 61402036).

References

Nikolaos Aletras and Mark Stevenson. 2013. Representing topics using images. In *Proceedings of NAACL-HLT*, pages 158–167.

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

A Elizabeth Cano, Yulan He, and Ruifeng Xu. 2014. Automatic labelling of topic models learned from twitter by summarisation. *ACL 2014*.

Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. 2012. Large-scale validation and analysis of interleaved search evaluation. *ACM Transactions on Information Systems (TOIS)*, 30(1):6.

Moses S Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388. ACM.

Aleksandr Chukllin, ANNE SCHUTH, KE ZHOU, and MAARTEN DE RIJKE. 2015. A comparative analysis of interleaving methods for aggregated search. *ACM Trans. Inf. Syst*, 33(2):5.

Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 253–262. ACM.

Matthew Hoffman, Francis R Bach, and David M Blei. 2010. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864.

Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2011. A probabilistic method for inferring preferences from clicks. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 249–258. ACM.

Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. 2012. Estimating interleaved comparison outcomes from historical click data. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1779–1783. ACM.

Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. 2013. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 465–474. ACM.

Zhipeng Jin, Qiudan Li, Can Wang, Daniel D Zeng, and Lei Wang. 2016. Labelling topics in weibo using word embedding and graph-based method. In *2016 International Conference on Information Systems Engineering (ICISE)*, pages 34–37. IEEE.

J.H. Lau, D. Newman, S. Karimi, and T. Baldwin. 2010. Best topic word selection for topic labelling. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 605–613. Association for Computational Linguistics.

J.H. Lau, K. Grieser, D. Newman, and T. Baldwin. 2011. Automatic labelling of topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1536–1545. Association for Computational Linguistics.

D. Magatti, S. Calegari, D. Ciucci, and F. Stella. 2009. Automatic labeling of topics. In *Intelligent Systems Design and Applications, 2009. ISDA'09. Ninth International Conference on*, pages 1227–1232. IEEE.

Gurmeet Singh Manku, Arvind Jain, and Anish Das Sarma. 2007. Detecting near-duplicates for web crawling. In *Proceedings of the 16th international conference on World Wide Web*, pages 141–150. ACM.

Xian-Ling Mao, Zhao-Yan Ming, Zheng-Jun Zha, Tat-Seng Chua, Hongfei Yan, and Xiaoming Li. 2012. Automatic labeling hierarchical topics. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2383–2386. ACM.

Yashar Mehdad, Giuseppe Carenini, Raymond T Ng, and Shafiq Joty. 2013. Towards topic labeling with phrase entailment and aggregation. In *Proceedings of NAACL-HLT*, pages 179–189.

Q. Mei and C.X. Zhai. 2005. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 198–207. ACM.

Q. Mei, C. Liu, H. Su, and C.X. Zhai. 2006. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *Proceedings of the 15th international conference on World Wide Web*, pages 533–542. ACM.

Q. Mei, X. Shen, and C.X. Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 490–499. ACM.

D. Ramage, P. Heymann, C.D. Manning, and H. Garcia-Molina. 2009. Clustering the tagged web. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 54–63. ACM.

D. Ramage, C.D. Manning, and S. Dumais. 2011. Partially labeled topic models for interpretable text mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–465. ACM.

Julia Vogel and Bernt Schiele. 2007. Semantic modeling of natural scenes for content-based image retrieval. *International Journal of Computer Vision*, 72(2):133–157.