

An Online Inference Algorithm for Labeled Latent Dirichlet Allocation

Qiang Zhou, Heyan Huang, and Xian-Ling Mao

Beijing Engineering Research Center of High Volume Language Information
Processing and Cloud Computing Applications,
Department of Computer Science and Technology,
Beijing Institute of Technology, Beijing 100081, China
{qzhou,hhy63,maoxl}@bit.edu.cn

Abstract. Using topic models to analyze documents is a popular method in text mining. Labeled Latent Dirichlet Allocation(Labeled LDA) is one of them that is widely used to model tagged documents and to solve relevant problems, such as tagged document visualization, snippet extraction and so on. However, traditional batch inference for Labeled LDA, which runs over entire document collection, is computationally expensive and not suitable for large scale corpora and text streams. In this paper, we develop an efficient online algorithm for Labeled LDA, called *online Labeled LDA*(online-LLDA). It is based on particle filter, a Sequential Monte Carlo approximation technique. Our experiments show that online-LLDA significantly outperforms batch algorithm(batch-LLDA) in time, while preserving equivalent quality.

Keywords: Online Inference, Online Labeled LDA, Particle Filter.

1 Introduction

In recent years, topic modeling has emerged as a relatively new approach to analyze text data. Lots of topic models, such as Probabilistic Latent Semantic Indexing(PLSI)[10], Latent Dirichlet Allocation(LDA)[3] and Hierarchical Dirichlet Processes(HDP) [18] have been proposed to model documents without tags.

Meanwhile, more and more pages in many websites, especially social media websites, are born with tags, or labels. These tags are usually assigned by users, and carry a lot of information about the pages. For example, a twitter user posts a tweet, and associates it with some hashtags, which produces a tagged document. Also, generally, category names of news pages can also be regarded as tags of pages.

Therefore, a number of topic models have been proposed to model labeled documents. Labeled LDA is one of them proposed by Ramage et al. in 2009, and has many extensions including PLDA[15], hLLDA[11] and so on. In Labeled LDA, a topic is defined as a word distribution, and each topic corresponds to a label. Thus, topics of each labeled document are restricted within its labels.

Also, through selecting the topic with highest probability for each word, Labeled LDA assigns the most appropriate label to each individual word. With this ability, it can solve many text analysis problems, such as credit attribution, tagged document visualization, snippet extraction and multi-labeled text classification[12][16]. For example, in tagged document visualization, annotating important words with their labels provides users a short semantic description of documents; In snippet extraction, by finding out snippet that contains most specific label, appropriate snippet can be extracted.

Traditional inference algorithms like Gibbs sampling and variational inference require multiple passes through the whole corpus. These algorithms are often referred to as *off-line* or *batch* algorithm. Obviously, it's difficult to apply batch-LLDA for large scale data. Considering the task of clustering all tagged web pages in *del.icio.us*, a popular social bookmarking website, it can be accomplished by using the topic distributions learned by Labeled LDA. Since it contains massive amounts of pages, and we have to run through all pages for many times, it will be time consuming and infeasible. Also, when documents arrive in a stream, batch algorithms do not work. For example, twitter users update status whenever they like to, so it will generate a flow of tweets. If we use batch algorithm to analyze these tweets, once a new tweet is available, we have to run the algorithm again over all tweets including previously seen ones to obtain the newest model parameters. That will be slower and slower as the number of tweets increases, which is unbearable in the real world. Thus, it's necessary to develop an efficient algorithm to solve the problem.

To the best of our knowledge, there is no existing work to obtain parameters of Labeled LDA by using an online style algorithm. Aiming at solving this problem, we propose an online algorithm, called online-LLDA, which incrementally updates model parameters when a document arrives. In this case, the algorithm do not need to visit the previous documents that have been processed. Consequently, these documents can be abandoned to save memory. Differently from batch Gibbs sampling, we use another sampling-based framework, particle filter[5], to make the algorithm work in online mode. We evaluate our method on two datasets, and show that online-LLDA performs better than batch-LLDA in time, while achieving equivalent effect.

The rest of the paper is organized as follows. We conclude the related work in Section 2. Our approach is introduced in Section 3. We demonstrate effectiveness and efficiency of our algorithm in section 4. The final conclusion and future work will be discussed in section 5.

2 Related Work

Most existing researches on LDA-like model utilize various inference algorithms, such as variational Bayesian[3][17], Gibbs sampling[7][8], expectation propagation [13] and belief propagation[21] to obtain the parameters. Unfortunately, most of them are batch algorithms.

Thus, a host of online learning methods have been developed for topic models. Some of them focus on modeling large amount of documents efficiently based

on LDA. Typical researches include TM-LDA[19], On-Line LDA[1] and so on. By minimizing the error between predicted topic distribution and the real distribution generated by LDA, TM-LDA captures the latent topic transitions in temporal documents. On-Line LDA uses topics learned from previous documents by LDA as the prior of following topics. It was designed by Alsumait et al.(2008) to detect and track topics in an online fashion.

Some other work try to improve inference algorithms themselves. Banerjee et al.(2007) presented online variants of vMF, EDCM and LDA. Their experiments illustrated faster speed and better performance on real-world streaming text[2]. Hoffman et al.(2010) developed an online variational Bayesian algorithm[9] for LDA based on online stochastic optimization. In their approach, they thought of LDA as a probabilistic factorization of the matrix of word counts into a matrix of topic weights and a dictionary of topics. Thus, they used online matrix factorization techniques to obtain an online fashion schema of LDA. Canini et al.(2009) also proposed an online version algorithm[4] using Gibbs sampling. Yao et al.(2009) compared several batch methods mentioned above, and introduced a new algorithm, SparseLDA[20] to accelerate learning process and consequently achieved nearly 20 times faster speed than traditional LDA.

To conclude, a large number of prior work have made great efforts on designing appropriate online algorithms for LDA. However, less work focus on improving the efficiency for Labeled LDA. In this paper, we propose a novel online learning method for Labeled LDA, called online-LLDA.

3 Method

In this section, we first review the Labeled LDA model along with its original batch algorithm. Then, we analyze the feasibility of applying particle filter on Labeled LDA in the following part. Finally, we present online-LLDA in detail.

3.1 Preliminary

Labeled LDA is a generative model for labeled documents that extends LDA. With the assumption that topics of labeled document are relevant with its labels, it adds a constraint on topic generation. Formally, for each document d in the collection of documents D , it can only pick up topics associated with its labels. As the graphical model shows in Figure 1, topics θ are determined by both topic prior and labels. Let K be the number of unique labels in all current observed documents, which equals the number of topics. Each document d has a label set $L_d = \{l_1^{(d)}, l_2^{(d)}, \dots, l_{K_d}^{(d)}\}$ where K_d is the number of labels. For the description convenience, $A_d = \{\lambda_1^{(d)}, \lambda_2^{(d)}, \dots, \lambda_{K_d}^{(d)}\}$ is defined to represent the label index set in document d . For each element $\lambda_j^{(d)}$ in A_d , we have $\lambda_j^{(d)} = \sum_{k=1}^K k \cdot I\{\lambda_j^{(d)} = l_k\}$, where I is the indicator function.

The original inference approach of Labeled LDA is a batch algorithm(batch-LLDA)[14], that iteratively samples topics of all words using the conditional

probability of latent variable z . For a word i , it is sampled according to the following equation:

$$P(z_i = k | \mathbf{z}_{\neg i}, \mathbf{w}) \propto I\{k \in \Lambda_d\} \cdot \left(\frac{n_{k, \neg i}^{(w_i)} + \beta}{n_{k, \neg i}^{(\cdot)} + V\beta} \cdot \frac{n_{d, \neg i}^{(k)} + \alpha}{n_{d, \neg i}^{(\cdot)} + K\alpha} \right) \quad (1)$$

where w_i is a word in document d , V is the size of vocabulary, K is the number of topics. $n_{k, \neg i}^{(w_i)}$ is the number of times word w_i is assigned to topic k , excluding w_i from all documents. $n_{k, \neg i}^{(\cdot)}$ is the corresponding summation over words. $n_{d, \neg i}^{(k)}$ is the count of words in document d with topic assignment k , excluding w_i . And $n_{d, \neg i}^{(\cdot)}$ is the corresponding summation over topics. α and β are prior parameters.

With Equation (1), batch Gibbs sampler can constantly sample topic for each word. After the burn-in period, topic assignments \mathbf{z} are available for estimation. To ensure the convergence, the number of iterations is usually set as big as enough. Hence, it will take too much time for batch-LLDA to reach the burn-in point, which leads to inefficiency.

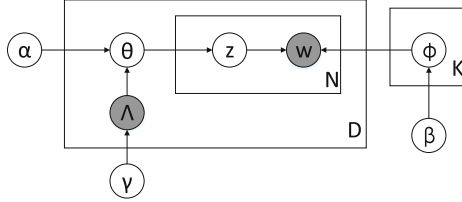


Fig. 1. Graphical model of Labeled LDA.

3.2 Feasibility Analysis

Labeled LDA can be viewed as a state space model, if we regard the latent variable topic z as a state variable, and the word w as an observation variable. Particle filter is a kind of efficient Markov-Chain Monte Carlo(MCMC) sampling method for estimating parameters of state space model. Differently from Gibbs sampling, it's an online algorithm.

Particle filter can be applied to solve the inference problem of Labeled LDA. (1) Suppose x is the hidden state variable and y is the observation variable in particle filter. The objective of particle filter is to estimate the values of hidden state variables given observations, that is $P(\mathbf{x}|\mathbf{y})$. Coincidentally, what we want to acquire in Labeled LDA is the joint distribution $P(\mathbf{z}|\mathbf{w})$, which is also the probability of state variables given observations. (2) Particle filter assumes that observations are conditionally independent, and observation y_k is only determined by x_k . Obviously, based on the bag-of-words assumption, Labeled LDA fulfills this requirement.

In this paper, we use Rao-Blackwellised particle filter(RBPF), an enhanced version of particle filter as our approximation method. RBPF integrates out the latent variables, and thus makes the solution simpler. In our algorithm, we have

P particles, and each particle p represents an answer that we desire, namely the posterior distributions of topics. In our implementation, a particle p stores all topic assignments of words, together with an importance weight $\omega^{(p)}$ indicating the importance of particle. What's more, a reassignment process is added to enhance the quality of samples.

3.3 Online Labeled LDA

In this section, we introduce an online learning method by using particle filter. As demonstrated in Algorithm 1, the overall algorithm consists of two phases: **initialization phase** and **online phase**. Initialization phase accomplishes the task of launching the online phase, while online phase continually processes every word w_i in a newly arrived document d and generates new parameter set Φ_d after the entire document has been processed.

In *initialization phase* (line 1 to line 4), for each particle, we apply batch-LLDA on an initial corpus E that contains a small fraction of documents. After running over, we get initial topic assignments of all initial words, along with sufficient statistics. These values are stored into each particle, which are useful in the online phase.

In *online phase* (line 5 to line 17), we first initialize particle weights with equal values and then process documents in a text stream one after another. Here, N_d represents the length of document d . Model parameters will be updated every time a document is processed. A new sampling equation is used as shown in Equation (2). In this equation, $\neg i$ has different meaning from Equation (1). In Equation (1), it represents all words or topics except i , but here it aims at excluding i from currently observed words. Thus, $\mathbf{i}\neg i$ represents first $i - 1$ words, $\mathbf{i}\neg j$ represents first i words except j . This difference is essential between batch-LLDA and online-LLDA. Also notice that when $i = 1$, $\mathbf{i}\neg i$ represents nothing, that is another reason why we need an initialization phase.

$$P(z_i^{(p)} = k | \mathbf{z}_{\mathbf{i}\neg i}^{(p)}, \mathbf{w}_i) \propto I\{k \in \Lambda_d\} \cdot \left(\frac{e_{k,w_i}^{(p)} + n_{k,\mathbf{i}\neg i}^{(w_i)(p)} + \beta}{e_{k,\cdot}^{(p)} + n_{k,\cdot}^{(\cdot)(p)} + V\beta} \cdot \frac{n_{d,\mathbf{i}\neg i}^{(k)(p)} + \alpha}{n_{d,\mathbf{i}\neg i}^{(\cdot)(p)} + T_d\alpha} \right) \quad (2)$$

Since we use all currently observed words including words in initial corpus, the word count in Equation (2) includes two parts. The first part is the word count of initial collection of documents, and the second part is the word count of documents coming in a stream in the online phase. The superscript p in all notations indicates the particle index. $e_{k,w_i}^{(p)}$ is the number of times word w_i is assigned to topic k in initial corpus, and $e_{k,\cdot}^{(p)}$ is the summation. n is similar with those in Equation (1). T_d is the number of unique labels when document d is available, namely the current topic number. All sampled topics should be restricted within its document label set.

In this algorithm, $P(z_i^{(p)} | \mathbf{z}_{\mathbf{i}\neg i}^{(p)}, \mathbf{w}_i)$ in Equation (2) is selected as the proposal distribution, so the importance weights are updated as $\hat{\omega}^{(p)} = \omega^{(p)} P(w_i | \mathbf{z}_{\mathbf{i}\neg i}^{(p)}, \mathbf{w}_i)$, and then normalized to sum to 1. $P(w_i | \mathbf{z}_{\mathbf{i}\neg i}^{(p)}, \mathbf{w}_i)$ is the probability of word i

Algorithm 1. Online Labeled LDA

Initialization Phase:

- 1 **for** $p = 1, \dots, P$ **do**
- 2 **while** *burn-in point is not reached* **do**
- 3 draw topic for each word w_i in initial corpus E , using Equation (1)
- 4 calculate sufficient statistics for each particle

Online Phase:

- 5 initialize importance weights $\omega^{(p)} = 1/P$ for any $p \in \{1, \dots, P\}$
- 6 **for** $d = 1, \dots, D$ **do**
- 7 **for** $i = 1, \dots, N_d$ **do**
- 8 **for** $p = 1, \dots, P$ **do**
- 9 draw sample $z_i^{(p)}$ using Equation (2)
- 10 $\hat{\omega}^{(p)} = \omega^{(p)} P(w_i | \mathbf{z}_{1-i}^{(p)}, \mathbf{w}_i)$
- 11 normalize $\omega^{(p)}$ for any $p \in \{1, \dots, P\}$, to sum to 1
- 12 calculate \hat{N}_{eff} using Equation (3)
- 13 **if** $\hat{N}_{eff} \leq N_{thresh}$ **then**
- 14 Sampling Importance Resample process
- 15 draw sample $z_r^{(p)}$ for any $r \in R(i)$, $p \in \{1, \dots, P\}$ using Equation (2)
- 16 set $\omega^{(p)} = 1/P$ for any $p \in \{1, \dots, P\}$
- 17 generate new parameter set Φ_d

in topic $z_i^{(p)}$, which is sampled in line 9. It is calculated using all currently sampled topic assignments, as the equation tells. Next, effective sample size \hat{N}_{eff} is calculated as:

$$\hat{N}_{eff} = 1 / \sum_{j=1}^P (\omega^{(j)})^2 \quad (3)$$

It is an estimation of N_{eff} , that measures the efficiency of the method and controls the algorithm to avoid degeneracy[6]. A sampling importance resample procedure (line 14) will be run if \hat{N}_{eff} is no more than threshold N_{thresh} . P particles are resampled with replacement according to the importance weights. Then old particles are replaced with the new ones. After this process, particles with small weight have high possibility to be eliminated. This process reflects the “survival of the fittest” law, “excellent” solutions should be inherited. Intuitively, N_{thresh} decides the frequency of resample, and thus influences the effectiveness and speed of the algorithm.

In addition, we add a reassignment period (line 15) after resample to improve the quality of samples. Since words coming in online phase are only sampled once, the result might be inaccurate. We solve this problem by picking up some words randomly, and reassigning topics to them. $R(i)$ is an index set, containing a fixed number of indexes that no more than i . These indexes are randomly selected from $\{1, \dots, i\}$, and represent the words reached earlier than word i including

word i itself. For each element r in $R(i)$, we sample a new topic according to Equation (2). Obviously, when $|R(i)|$ is big enough, online-LLDA will degenerate to a batch algorithm, since previous words will be reassigned constantly.

Generally, our final objective, the joint distribution $P(\mathbf{z}|\mathbf{w})$ as we mentioned in section 3.2 is approximated as below:

$$P(\mathbf{z}|\mathbf{w}) \approx \sum_{p=1}^P \omega^{(p)} I(\mathbf{z}, \mathbf{z}^{(p)}) \quad (4)$$

where $I(\mathbf{z}, \mathbf{z}^{(p)})$ is an indicator function,

$$I(\mathbf{z}, \mathbf{z}^{(p)}) = \begin{cases} 1 & \text{if } \mathbf{z} \text{ equals } \mathbf{z}^{(p)} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

In other words, particles with same vector \mathbf{z} , have same $P(\mathbf{z}|\mathbf{w})$. And it equals the sum of weights of these particles. Then, topic assignments \mathbf{z}^* for parameter estimation are calculated as follows:

$$\mathbf{z}^* = \arg \max_{\mathbf{z}^{(p)}} P(\mathbf{z}^{(p)}|\mathbf{w}) \quad (6)$$

However, in reality, we found that it cost too much time to check two particles whether they share the same \mathbf{z} , since \mathbf{z} is a vector whose length is the size of all words. We also found that there were seldom same particles. Therefore, we modify this procedure to choose the particle with biggest weight for estimation. The modified \mathbf{z}^* is:

$$\mathbf{z}^* = \mathbf{z}^{(p^*)}, \quad p^* = \arg \max_p \omega^{(p)} \quad (7)$$

With \mathbf{z}^* , we can compute word count and estimate parameters we need. After document d is processed we can get Φ_d . It contains word distributions, or topics for all labels.

4 Experiment

We evaluated both efficiency and effectiveness of our method on two real datasets through three experiments. The first one compares the interpretability of two algorithms. The second one shows the effectiveness of batch-LLDA and online-LLDA, by using perplexity as metric. And the last one demonstrates the power of our online-LLDA in fast processing document.

4.1 Experiment Setting

The first dataset is a collection of papers crawled from the ACM website¹, which consists of 1712 full papers of four conferences (CIKM, SIGIR, SIGKDD

¹ <http://dl.acm.org/>

and WWW) from the year 2011 to the year 2013. We refer to this corpus as **Conf** in the rest of this paper. We used keywords in the papers as labels. The second dataset, called **Twitter**, is a corpus of tweets downloaded from Twitter, which contains about 2 million tweets. We used hashtags as labels in this corpus. The detailed information of datasets is listed in Table 1. All experiments were run on a server with an Intel Xeon E3-1230 3.3 GHz CPU and 32GB memory.

Table 1. Dataset Description

Dataset	Conf	Twitter
Document Size	1,712	2,180,548
Number of Labels	3,537	5,615
Vocabulary Size	17,697	14,977

In all of our experiments, α and β are fixed at 0.1, and burn-in iteration time is 1000 for batch-LLDA. Since $|R(i)|$, N_{thresh} and P influences the effectiveness and runtime of online-LLDA, we should make a tradeoff. In our experiments, $|R(i)|$ is set as 30 for the smaller dataset Conf and 10 for Twitter, N_{thresh} is 1.5, and P is 10.

4.2 Topic Visualization

In this experiment, we ran both batch-LLDA and online-LLDA over two entire corpora. Then, we picked up top ten words of each topic learned from each algorithm to represent a topic. Each topic is associated with its corresponding label. Table 2 compares the results of two algorithms on **Conf** corpus, while Table 3 shows the **Twitter**’s.

Most topics generated by our online-LLDA are as interpretable as batch-LLDA. And these topics are highly relevant with corresponding labels. For example, in Conf, the top words “topic”, “topics” and “modeling” explain the label “topic modeling” well; “user”, “item” and “recommendation” are frequently used to describe “recommender systems”; In Twitter, when talking about “politics”, people are most likely to discuss “found”, “obama” and “health”.

4.3 Document Modeling

In computational linguistics, perplexity is a widely used metric that measures how well a language model predicts words in the test corpus. Lower perplexity indicates higher likelihood of the test corpus, or better generation performance of the model. Generally, it is computed as Equation (8). Since topics in Labeled LDA are relevant with labels, when a label in the test corpus do not appear in the training corpus, the algorithm will not assign any topics that relates with this label on words. Thus, only prior parameter α can influence the word distribution θ in this case. It means that, in this topic, all words share the same probability to

Table 2. Top ten words from five topics with labels learned in the **Conf** corpus.

Label	Top Ten Words	
	batch-LLDA	online-LLDA
topic modeling	topic, topics, model, document, word, words, number, segment, lda, distribution	topic, topics, modeling, entity, dirichlet, mining, lda, distributions, name, text
deep learning	semantic, gram, embedding, deep, training, latent, vocabulary, layer, vector, hashing	deep, hashing, letter, layer, neural, gram, unsupervised, speech, layers, dimensionality
recommender systems	user, users, item, recommendation, model, based, data, items, recommender, systems	user, item, recommendation, items, data, model, ratings, recommender, system, rating
sentiment analysis	sentiment, set, based, analysis, opinion, words, approach, data, results, pages	sentiment, opinion, based, analysis, classification, pages, performance, data, using, approach
svm	classification, sampling, data, methods, results, sentiment, class, learning, svm, tweets	methods, svm, time, performance, optimization, problem, table, accuracy, words, training

Table 3. Top ten words from five topics with labels learned in the **Twitter** corpus.

Label	Top Ten Words	
	batch-LLDA	online-LLDA
apple	apple, iphone, ipod, new, itunes, event, touch, via, snow, nano	apple, ipod, iphone, new, itunes, event, snow, touch, via, leopard
dogs	dogs, dog, new, please, korean, million, animals, pls, news, too	dogs, dog, pets, update, pet, animals, new, cats, please, pls
gift	gift, super, shop, album, handmade, rock, art, music, cover, recycled	gift, perfect, lover, consider, friend, love, store, jacket, any, friends
photography	photography, photo, photos, new, via, digital, camera, how, photographer, get	photography, photo, new, photos, via, digital, how, camera, get, blog
politics	politics, found, obama, news, health, speech, us, obamas, care, glenn	politics, found, obama, health, speech, care, obamas, glenn, beck, us

be generated, which leads to low ability of predicting words in unseen documents and high perplexity.

$$perplexity(D_{test}) = \exp \left\{ - \frac{\sum_{d=1}^M \log p(\mathbf{w}_d)}{\sum_{d=1}^M N_d} \right\} \quad (8)$$

In this experiment, we simulated the situation that documents are coming in a stream. In our online-LLDA, model parameters are updated every time a new document arrives. We use 200 documents in Conf and 5% tweets in Twitter for initialization. For the sake of fairness, we excluded topics that were sampled in

initialization phase when generating the model. We computed perplexity of the held-out test dataset at some points using the current model learned by online-LLDA. As for batch-LLDA, we computed perplexity at the same points, however each run has to use all of the documents previously seen.

Figure (2) shows that, in both corpora, perplexity is much higher at the start, and then declines as the seen documents number increases. It is reasonable since the seen documents used for training are not enough at first, and lack most of labels in test corpus. As the training dataset grows, the learned model fits the test corpus better, and perplexity converges. In Conf, we achieved better effect than batch-LLDA. In Twitter, the two curves are very close, and online-LLDA's is lower than batch-LLDA's as we can see in the zoomed fragment. It's not surprise since online-LLDA incorporates a resample process and a reassignment period. Also, better particles will remain according to the algorithm, which leads to better effectiveness. Since the number of tweets is large, and a single tweet is short, the result in Twitter is satisfactory. Notice that the document numbers in Twitter are in logarithmic scale.

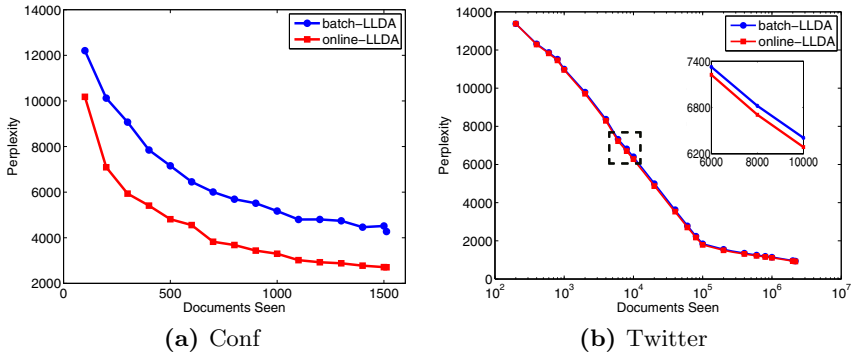


Fig. 2. Perplexity On Two Datasets

4.4 Efficiency

In this experiment, we evaluated the efficiency for each algorithm, which is the main purpose of online-LLDA. We used the same points as we described in section 4.3, and recorded the training time using current observed documents.

When new documents arrive, batch-LLDA has to run over all documents for many iterations again. Since the time for each iteration grows with the number of documents, the total time for batch-LLDA grows fast. However, online-LLDA does not need to do this, it should only process the new coming document and update parameters to get a new model. As Figure (3) shows, batch-LLDA costs much more time to get the new parameters compared with online-LLDA, especially when the number of observed documents is large. When running over the Conf corpus, the training time of online-LLDA is less than 400 seconds, while batch-LLDA takes more than 3,000 seconds, which is about 8 times longer. In twitter, batch-LLDA takes more than 10,000 seconds, while online-LLDA takes less than 6,000 seconds.

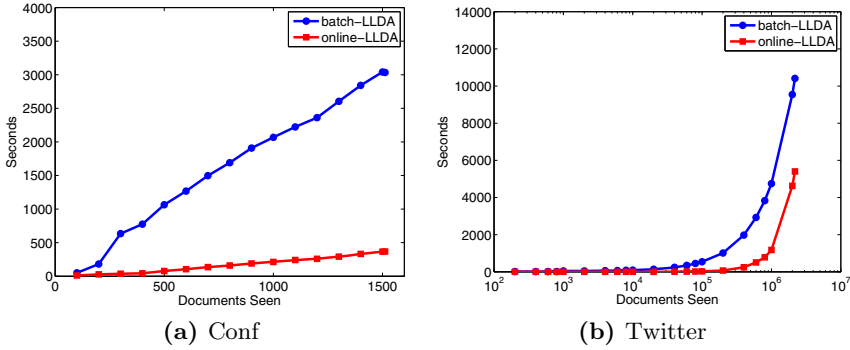


Fig. 3. Training Time On Two Datasets

5 Conclusion and Future Work

In this paper, we proposed an online method for Labeled LDA inference, called online-LLDA. We analyzed the feasibility of applying particle filter, a MCMC method on Labeled LDA and demonstrated that it's a feasible choice for Labeled LDA. We presented our algorithm in detail and conducted several experiments on two real datasets. Our experiment results clearly show that online-LLDA performs as good as batch-LLDA, and costs much less time.

In the future, we will explore the following directions. (1) We will speed up this algorithm through optimizing the data structure of particle and word count. (2) We will try other types of method, including online version of variational Bayesian, expectation propagation, belief propagation, and so on.

Acknowledgment. The work was supported by National Natural Science Foundation of China (Grant No. 61402036 and No. 3070021501109), and 863 Program of China (Grant No. 2015AA015404).

References

1. AlSumait, L., Barbará, D., Domeniconi, C.: On-line lda: Adaptive topic models for mining text streams with applications to topic detection and tracking. In: ICDM 2008, pp. 3–12. IEEE (2008)
2. Banerjee, A., Basu, S.: Topic models over text streams: A study of batch and online unsupervised learning. In: SDM, vol. 7, pp. 437–442. SIAM (2007)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research 3, 993–1022 (2003)
4. Canini, K.R., Shi, L., Griffiths, T.L.: Online inference of topics with latent dirichlet allocation. In: International Conference on Artificial Intelligence and Statistics, pp. 65–72 (2009)
5. Doucet, A., De Freitas, N., Murphy, K., Russell, S.: Rao-blackwellised particle filtering for dynamic bayesian networks. In: UAI 2000, pp. 176–183. Morgan Kaufmann Publishers Inc. (2000)
6. Doucet, A., Godsill, S., Andrieu, C.: On sequential monte carlo sampling methods for bayesian filtering. Statistics and Computing 10(3), 197–208 (2000)

7. Griffiths, T.L., Steyvers, M.: Finding scientific topics. *PNAS* 101(suppl. 1), 5228–5235 (2004)
8. Heinrich, G.: Parameter estimation for text analysis. Tech. rep. (2005)
9. Hoffman, M., Bach, F.R., Blei, D.M.: Online learning for latent dirichlet allocation. In: *NIPS 2010*, pp. 856–864 (2010)
10. Hofmann, T.: Probabilistic latent semantic indexing. In: *SIGIR 1999*, pp. 50–57. ACM (1999)
11. Kang, D., Park, Y., Chari, S.N.: Hetero-labeled LDA: A partially supervised topic model with heterogeneous labels. In: Calders, T., Esposito, F., Hüllermeier, E., Meo, R. (eds.) *ECML PKDD 2014, Part I. LNCS*, vol. 8724, pp. 640–655. Springer, Heidelberg (2014)
12. Li, X., Ouyang, J., Zhou, X.: Supervised topic models for multi-label classification. *Neurocomputing* (2014)
13. Minka, T., Lafferty, J.: Expectation-propagation for the generative aspect model. In: *UAI 2002*, pp. 352–359. Morgan Kaufmann Publishers Inc. (2002)
14. Ramage, D., Hall, D., Nallapati, R., Manning, C.D.: Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In: *EMNLP 2009*, pp. 248–256. Association for Computational Linguistics (2009)
15. Ramage, D., Manning, C.D., Dumais, S.: Partially labeled topic models for interpretable text mining. In: *KDD 2011*, pp. 457–465. ACM (2011)
16. Rubin, T.N., Chambers, A., Smyth, P., Steyvers, M.: Statistical topic models for multi-label document classification. *Machine Learning* 88(1-2), 157–208 (2012)
17. Teh, Y.W., Newman, D., Welling, M.: A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In: *NIPS 2006*, pp. 1353–1360 (2006)
18. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical dirichlet processes. *Journal of the American Statistical Association* 101(476) (2006)
19. Wang, Y., Agichtein, E., Benzi, M.: Tm-lda: efficient online modeling of latent topic transitions in social media. In: *KDD 2012*, pp. 123–131. ACM (2012)
20. Yao, L., Mimno, D., McCallum, A.: Efficient methods for topic model inference on streaming document collections. In: *KDD 2009*, pp. 937–946. ACM (2009)
21. Zeng, J., Cheung, W.K., Liu, J.: Learning topic models by belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35(5), 1121–1134 (2013)

Web Technologies and Applications

17th Asia-Pacific Web Conference, APWeb 2015,

Guangzhou, China, September 18-20, 2015,

Proceedings

Cheng, R.; Cui, B.; Zhang, Z.; Cai, R.; Xu, J. (Eds.)

2015, XIX, 885 p. 330 illus., Softcover

ISBN: 978-3-319-25254-4