Online Topic Exploration of Document Collections

No Author Given

No Institute Given

Abstract. When users perform online exploratory data analysis, they usually need insights about current data to decide next action. Probabilistic topic models can help us discover underlying thematic structure in large document collections without reading each of the documents. This paper presents a new approach for constructing topic structures in document subsets. With the aid of global precomputation, our algorithm is efficient enough for online processing, while remaining comparable performance in topic modeling to standard topic model. Besides, the proposed algorithm can be easily extended to a parallel one for further acceleration.

1 Introduction

With data explosion in Internet age, it's challenging for users to extract useful information from vast amounts of data to help them make decisions. The difficulty becomes rather pronounced in interactive data analysis, where users are often not clear whether a collection of data contains information of their interests. Usually, lots of time is spent in browsing records in current collection. Sometimes, users move to another collection before browsing important information. This type of problem can be solved with providing users an overall insights about the presented collection of data. Take the numerical data as example, if the users are presented with average, maximum, minimum and other statistics, they will have a basic understanding of whether a collection contains records they are interested in.

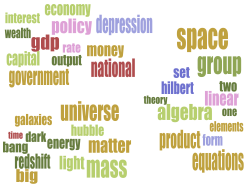
Unstructured text data has such a fast mode of transmission, and it's very time consuming for users to read a collection of documents, thus tools helping users understand what the document collections talks about become very useful. Word cloud is a widely used approach to do this task. By visualizing and highlighting popular terms based on occurrence frequency and prominence, word cloud communicates much information in a single glance. Fig.1a shows a word cloud demo of a document set, from which we can easily figure out it's a collection about cosmic physics.

However, when collection consists of documents from different domains, the representation generated by word cloud may be a little bit confusing. Fig.1b illustrates such a case, the presented words seems selected randomly (though actually not) from dictionary, you can hardly understand what the collection talks

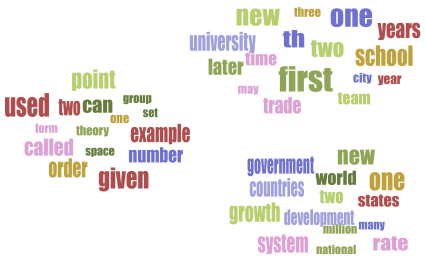
about. This limitation of simply highlighting words based on occurrence frequency raise the needs of approach that can discover the hidden topic structures of text collections.

(a) word cloud

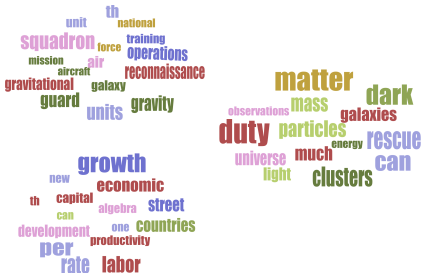
(b) word cloud for documents from different domains



(c) topic model representation



(d) representation for baseline method 1



(e) representation for baseline method 2

Fig. 1: Representations generated by various methods.

Probabilistic topic models[3][16][14] are techniques for summarizing the content of large text corpora. These techniques model topics as distributions of words, and represent each document as a combination of these topics. The extracted topics summarize the corpus and pass refined information to users. And for most topics the distributions of words tend to be sparse, so a good representation of topic is as a set of words that have high probability. Fig.1c demonstrates the output of a topic model trained on previous document collection.

The constructed topic structure enable user to do analysis processing in topic dimension. User can get coarse-grained or fine-grained topic information by adjusting the number of topics in target set. And if a particular topic attracts the user, the documents related to that topic could be retrieved and displayed in the order of relevancy, allowing user to browser the specific documents one by one.

Although topic model provides a solution to depict document set, it is not practical for online analysis processing. For a subset with moderate size (e.g. 2000 documents, 300,000 tokens), training a topic model from scratch may take several minutes, which is intolerant for online users. It's easy to come up with some intuitive methods to avoid retraining from scratch. For instance, we can first train a large topic model on whole corpus ahead of time, meanwhile, we store the topics proportion of each document. When users request a subset of docu-

ments, we then present top topics extracted from corpus based on corresponding proportions in the target subset. This method could be extremely efficient, but lacking in capacity of summarizing target documents accurately. Fig.1d gives an illustration. There are two main reasons for this unsatisfactory performance, one is that the topics trained on whole corpus are too general for a subset, the subset-specific prominent words are ignored in this representation. Another is that the distribution of corpus-level topics in a subset tends to be even, thus the presented top topics only cover a small part of the collection. With these defects, this baseline method performs poorly in subset modeling, the quantitative results will be demonstrated in Section5.1.

Another intuitive method is to cluster documents in topic space first, and then use the most frequent words in each cluster to represent the cluster (Fig.1e). Although this approach is efficient enough and capable for discovering subset-specific prominent words, its representation for each cluster is not topic coherent. This is because (1) clustering process can hardly classify each document correctly, and (2) each document itself is a combination of multiple topics, hence the clustered documents are not topic coherent.

In this paper, we study the problem of subset topic modeling in the case of online processing, and propose a feasible solution. The main contributions of this paper are summarized as follows:

- This paper proposes OLES, an efficient algorithm for constructing topic structures in document subset. Experimental results show that OLES outperforms standard topic model up to 200 times in efficiency.
- We present two baseline methods, corpus-level topic selection based method (baseline1) and document-clustering based method (baseline2) in Section4.1.
- We evaluate the performance of each method in subset modeling on two real-world datasets, the results show that OLES achieves comparable performance to standard topic model.
- We extend the proposed algorithm to a parallel version, which obtains a further acceleration.

This paper begins with a review of the relevant literature (Section2), which is followed by a brief discussion of LDA (Section3). We then introduce the proposed algorithm and several baseline methods for subset modeling in Section4, and demonstrate corresponding experiments in Section5.

2 Related Works

Latent Dirichlet Allocation[3] is a typical unsupervised method for performing probabilistic topic modeling, and has been widely used in industrial application such as information retrieval and text classification. As a generative model, LDA assumes that the words in a document are generated by first choosing a latent topic for each word, then using the topic to generate the word itself (according to the topic's multinomial distribution over words). Given the generative model and the document corpus, the hidden topics and the proportions of topics in

each document can be estimated by inference method. We will introduce the details of generative process and the inference method of LDA in Section3.

There are several works based on topic modeling aiming to help users understand and navigate large document collection. Exemplar-based visualization[7] presents a probabilistic multidimensional projection model in the low-rank text subspace, and uses the selected exemplars to represent and visualize the text corpus. Topic browser[9] implements an web-based interactive browser to display the output of topic models, FacetAtlas[5] proposes a multifaceted visualization technique which combines search technology and advanced visual analytical tools. The tools mentioned above help users understand the corpus as a whole, while some studies[6][10][15] focus on providing both a high-level summary of the corpus and links between the summary and individual documents.

Researches about hierarchical topic model have been conducted, such as the Pachinko Allocation[17], nested Chinese restaurant process[2] and DCM-LDA[18]. There also exists methods which create topic visualizations of a documents collection hierarchically[21][8]. These tools make it possible for users to do interactive topic analysis by ways of drilling down or scrolling.

However, all these methods prepare representation for a specific collection. Once the requested set of documents changes, we need to do heavy recomputation from scratch, which is very inconvenient for online exploratory analysis. To meet the performance requirement for online processing, a topic modeling algorithm with higher efficency is indispensable.

Studies abound on promoting efficiency for topic model. Some focus on distributed algorithms for topic models[19][13][23]. Some are concerned with how to run an online learning algorithm in the case of corpus growing over time[12][4].Some concentrate on accelerating topic model inference on streaming document[22][1]. They do improve the efficiency, however, they still need training from scratch once the collection changes, thus the time cost can't meet the demand for making topic representation instantly for an arbitrary collection. In this work, we present a novel algorithm to solve this problem. In following sections, we will introduce the algorithm in detail and demonstrate corresponding experiments.

3 Latent Dirichlet Allocation

This section starts by reviewing the probabilistic topic model, focusing on Latent Dirichlet Allocation (LDA). As a generative model, the LDA model can be divided into two parts, probabilistic generative process and inference algorithm.

3.1 Probabilistic Generative Process

LDA models each of the M documents in collection as a mixture over K latent topics $\Phi = \varphi_{1:K}$, with each topic φ_k is a multinomial distribution over the vocabulary of T words. The generative processing of LDA works as follows,

1 For each of the K topics, draw φ_k from a Dirichlet prior,
   $\varphi_k \sim Dirichlet(\beta)$
2 For each of the M documents, draw topic proportion $\theta_m \sim Dirichlet(\alpha)$
3 For each word i in document m,
   (1) Draw topic assignm $z_{mi} \sim Multinomial(\theta_m)$.
   (2) Draw word token $w_{mi} \sim Multinomial(\varphi_{z_{mi}})$.
   Fig.2 demonstrates the graphical model representation of LDA,

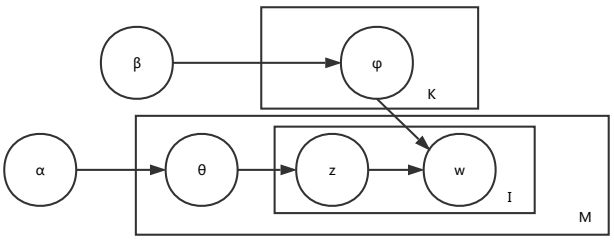


Fig. 2: Graphical model for LDA

3.2 Inference Algorithm

Given the generative model and a certain set of documents W_{mi}, the core of LDA is approximating the posterior distribution of latent random variables. A popular method for inference is gibbs sampling[11], which is based on Markov chain Monte Carlo (MCMC),

1 For each word w_{mi} in each document, randomly assign a topic number z_{mi}.
2 Re-scan the corpus and use gibbs sampling to re-sample each word's topic z_{mi}.
3 Repeat step2 until convergence.
4 With flagged words, calculate the estimation of Φ and θ.

Griffiths[11] gives the formula for determining topic tokens (Equation1), which is used in step2, and the algorithm for inferring Φ and θ (Equation2, Equation3), which is used in step4,

$$p(z_i = k|\overrightarrow{\mathbf{z}}_{\neg i}, \overrightarrow{\mathbf{w}}) \propto \frac{n_{mk,\neg i} + \alpha_k}{\sum_{k=1}^{K}(n_{mk,\neg i} + \alpha_k)} \cdot \frac{n_{kt,\neg i} + \beta_t}{\sum_{t=1}^{T}(n_{kt,\neg i} + \beta_t)} \tag{1}$$

$$\hat{\theta}_{mk} = \frac{n_{mk} + \alpha_k}{\sum_{k=1}^{K}(n_{mk} + \alpha_k)} \tag{2}$$

$$\hat{\varphi}_{kt} = \frac{n_{kt} + \beta_t}{\sum_{t=1}^{T}(n_{kt} + \beta_t)} \tag{3}$$

For ease of reading, Table1 lists the descriptions of commonly used variables.

Table 1: Description of commonly used variables

Notation	Description
M	Number of documents in corpus
T	Number of distinct words in vocabulary
K	Number of topics
α, β	Prior parameters for dirichlet distribution
w_{mi}	ith word in document m
z_{mi}	Topic assigned to word w_{mi}
φ_k	Distribution over vocabulary of topic k
θ_m	Distribution over topics of document m
n_{mk}	Count of topic k assigned in document m
n_{kt}	Count of word t assigned to topic k
$subM$	Number of documents in subset
$subK$	Number of topics in subset

4 Topic Exploration of Document Subset

4.1 Baseline Methods

As discussed in section1, it's time consuming to train LDA model. For online analysis, running a complete LDA model from scratch for a certain subset can not meet the time requirement. There is a need for methods which can immediately construct the topic structures of a document subset. In introduction, we mentioned two intuitive methods that avoid re-training in subset through pre-training in corpus, baseline1 and baseline2. Here we give the mathematical expressions of two methods.

Baseline1 is to select corpus-level topics with the highest weight in target subset. For a subset with $subM$ documents $\{\overrightarrow{w}_1^{sub}, \overrightarrow{w}_2^{sub}, \ldots, \overrightarrow{w}_{subM}^{sub}\}$, select the most dominant topics as representation. Algorithm1 gives the details,

Algorithm 1 baseline1 for topic modeling in subsets

1: Weight the topic distributions over documents $\{\overrightarrow{\theta}_1^{sub}, \overrightarrow{\theta}_2^{sub}, \ldots, \overrightarrow{\theta}_{subM}^{sub}\}$ by document length, get the topic distribution $\overrightarrow{\theta}^{sub}$ over target subset.
2: Based on $\overrightarrow{\theta}^{sub}$, choose $subK$ topics $\{\overrightarrow{\varphi}_1^{sub}, \overrightarrow{\varphi}_2^{sub}, \ldots, \overrightarrow{\varphi}_{subK}^{sub}\}$ with highest weight.
3: **return** $\{\overrightarrow{\varphi}_1^{sub}, \overrightarrow{\varphi}_2^{sub}, \ldots, \overrightarrow{\varphi}_{subK}^{sub}\}$ as the topics of target subset.

In baseline1, the selected topics only cover a part of the subset, and are too general to demonstrate subset-specific information. Baseline2 eliminates these defects. By clustering documents into $subK$ clusters in topic space, baseline2 assigns documents with similar topics into the same cluster. Then word distribution of each cluster is taken as the cluster's topic, the $subK$ topics produced in this way are selected as topics of target subset.

Algorithm 2 baseline2 for topic modeling in subsets

1: With $\{\overrightarrow{\theta}_1^{sub}, \overrightarrow{\theta}_2^{sub}, \ldots, \overrightarrow{\theta}_{subM}^{sub}\}$, map documents into $subM$ points in K-dimensional topic space. Run K-means on these points to cluster them into $subK$ clusters.
2: For kth cluster in $subK$ clusters, use its word distribution as kth topic $\overrightarrow{\varphi}_k^{sub}$ of the subset.
3: **return** $\{\overrightarrow{\varphi}_1^{sub}, \overrightarrow{\varphi}_2^{sub}, \ldots, \overrightarrow{\varphi}_{subK}^{sub}\}$ as topics of target subset.

However, it's hard to cluster each document into correct group. And a document itself is actually a mix of multiple topics, thus a cluster consists of multiple documents could hardly be topic consistent, which makes the representation hard to understand. The experimental results will give an illustration.

4.2 Proposed Algorithm

Inspired by the drawbacks of these two baseline approaches, we focus on designing OLES, an improved method to demonstrate subset-specific information, which also covers all documents and allow each document to contain multiple topics.

In addition to documents' topic proportion $\overrightarrow{\theta}$, we can go further to save all assigned topic tokens of words in model training. The distribution of words in target subset flagged as topic k over vocabulary, denoted as $\varphi_k^{'}$, can be treated as a minor topic. On one hand, the dominant words in $\varphi_k^{'}$ are extracted from target subset, thereby maintain subset-specific information. On the other hand, these words are selected from a well pre-trained corpus level topic, which guarantees the topic coherence.

With quite a large number of minor topics, we need to merge these topics into $subK$ "larger" topics, and each of the $subK$ topics should be topic coherent (one may argue that, just simply listing all minor topics also produces a reasonable representation. However, since there may be thousands of minor topics, this representation is nearly difficult and time consuming to understand as reading each of the documents themselves). Some statistical distance metric (K-L divergence, Hellinger distance, etc) could be used in topic merging. The complete algorithm works as follows,

Algorithm 3 OLES algorithm for topic modeling in subsets

1: With word tokens of $subM$ documents in target subset $\{\overrightarrow{w}_1^{sub}, \overrightarrow{w}_2^{sub}, \ldots, \overrightarrow{w}_{subM}^{sub}\}$ and corresponding topic tokens $\{\overrightarrow{z}_1^{sub}, \overrightarrow{z}_2^{sub}, \ldots, \overrightarrow{z}_{subM}^{sub}\}$, count the number of word t flagged as topic k in target subset $N_{kt}^{(sub)}$ to get K minor topics $\{\overrightarrow{\varphi}_1^{'}, \overrightarrow{\varphi}_2^{'}, \ldots, \overrightarrow{\varphi}_K^{'}\}$.
2: Choose $subK$ topics from K minor topics $\{\overrightarrow{\varphi}_1^{'}, \overrightarrow{\varphi}_2^{'}, \ldots, \overrightarrow{\varphi}_K^{'}\}$ as the initial centers of $subK$ clusters.
3: Assign each of K minor topics to the cluster whose center is closest to the minor topic.
4: Merge minor topics in each cluster as this cluster's topic $\overrightarrow{\varphi}_{subk}^{sub}$, and take the merged topic as new center of this cluster.
5: Repeat step3-4 until convergence.
6: **return** $\{\overrightarrow{\varphi}_1^{sub}, \overrightarrow{\varphi}_2^{sub}, \ldots, \overrightarrow{\varphi}_{subK}^{sub}\}$ as the topics of target subset.

As with common clustering algorithms (Kmean, Kmeans++, DBSCAN, etc), there are several options for initial center selection (step2) and distance measure (step3). Here we use minor topics with maximum counts as initial centers and Hellinger distance as distance measure, since in this way the algorithm gives best experimental results.

Algorithm 4 parallelized OLES algorithm

1: Assign $subM$ documents to P processors.
2: **for** p in P processors **do**
3: Count the number of word t flagged as topic k in processor p locally, save the numbers in sparse matrix $N_{ktp}^{(sub)}$.
4: Report $N_{ktp}^{(sub)}$ to master node.
5: **end for**
6: Add up $N_{ktp}^{(sub)}$ to get $N_{kt}^{(sub)}$, use $N_{kt}^{(sub)}$ to estimate the K minor topics in subset $\{\overrightarrow{\varphi}_1^{'}, \overrightarrow{\varphi}_2^{'}, \ldots, \overrightarrow{\varphi}_K^{'}\}$.
7: Choose $subK$ topics from K minor topics $\{\overrightarrow{\varphi}_1^{'}, \overrightarrow{\varphi}_2^{'}, \ldots, \overrightarrow{\varphi}_K^{'}\}$ as the initial centers of $subK$ clusters.
8: Assign K minor topics to P processors.
9: Broadcast centers.
10: **for** p in P processors **do**
11: **for** k_p in K_p minor topics assigned to processor p **do**
12: Assign topic k_p to the cluster $subk$ whose center is closest to topic k_p.
13: **end for**
14: Report $(k_p, subk)$ pairs to master node.
15: **end for**
16: Merge minor topics in each cluster as this cluster's topic $\overrightarrow{\varphi}_{subk}^{sub}$, and take the merged topic as new center of this cluster.
17: Repeat step9-16 until convergence.
18: Return $\{\overrightarrow{\varphi}_1^{sub}, \overrightarrow{\varphi}_2^{sub}, \ldots, \overrightarrow{\varphi}_{subK}^{sub}\}$ as the topics of target subset.

Parallelization Issue The OLES algorithm can be parallelized for a further acceleration. Both time-comsuming parts of the algorithm, generating and merging the minor topics, can be generalized to a parallel version. Algorithm.4 describes the parallel algorithm in detail.

5 Experiments

5.1 Perplexity

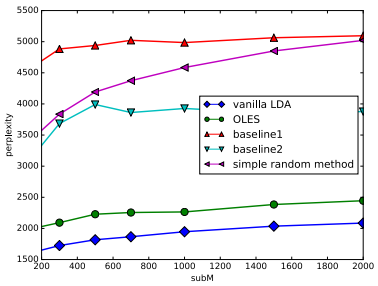
We experiment on two datasets, Enron Email and one million wikipedia articles, here we use *perplexity* to evaluate the performance of model fit. Perplexity is a widely used measure in topic modeling, for a test set with $subM$ documents, the perplexity is,

$$perplexity(D) = \exp\{-\frac{\sum_{m=1}^{subM} \log p(\overrightarrow{w}_d)}{\sum_{m=1}^{subM} N_m}\} \tag{4}$$

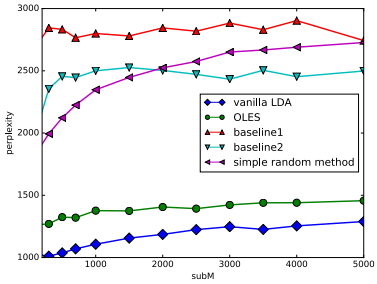
Taking into account the subset is generally small, if randomly split documents into train set and test set, the underlying topics of two sets may be of huge difference. In this case, perplexity on held-out test set is probablely not a good indicator for modeling performance. To prevent this, we randomly divide each document into two parts, 80% of words are selected as a document in train set and 20% in test set. With this approach, the underlying topics of test set and train set don't differ greatly.

Fig.3a and Fig.3b show different methods' perplexity with different $subM$ (number of documents in subset), every point in figure is the average perplexity value of 50 randomly selected subsets in corresponding condition. To highlight the performance of proposed method in subset modeling, we add extra perplexity results from an simple random method. The simple method randomly assign words in documents to $subK$ topics and take them as the topic structure of target subset.

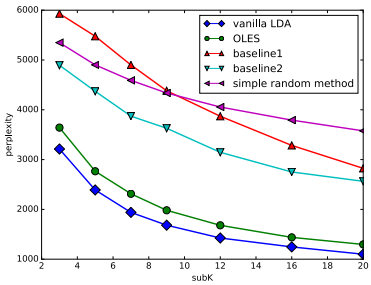
Fig.3c and Fig.3d display different methods' perplexity with different $subK$ (number of topics in subset). Considering that our task is to make a representation to reduce the time users browse records in collection, in this scenario the value of $subK$ shouldn't be too large.



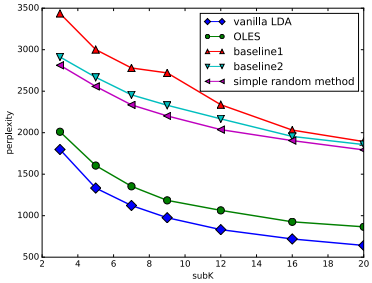
(a) Average perplexity of 50 runs on one million wikipedia articles (1000,000 docs, 200,000,000 tokens, 30,000 vocabulary size, 300 corpus-level pre-trained topics, subK=7)



(b) Average perplexity of 50 runs on Enron-Email dataset (500,000 docs, 80,000,000 tokens, 30,000 vocabulary size, 200 corpus-level pre-trained topics, subK=7)



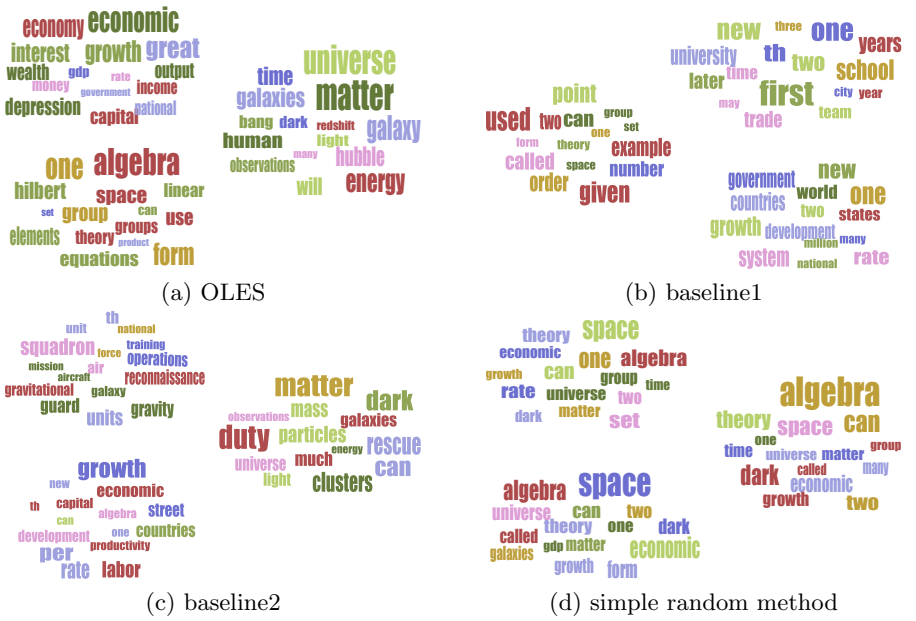
(c) Average perplexity of 50 runs on one million wikipedia articles, with subM=1000



(d) Average perplexity of 50 runs on Enron-Email dataset, with subM=1000

Fig. 3: Comparison of the perplexity results of five methods for subset modeling: re-training a LDA model in subset from scratch (vanilla LDA), OLES, baseline1, baseline2, randomly assigning words in subset to *subK* topics (simple random method).

In all conditions, experimental results show that our algorithm performs far more better than all baselines in modeling perplexity, approximating to standard LDA.



(a) OLES (b) baseline1

(c) baseline2 (d) simple random method

Fig. 4: Comparison of human readability

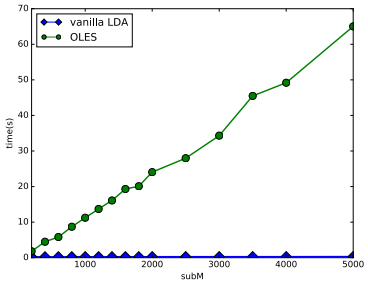
Fig.4 illustrates representations given by different methods for a collection, from which we can see that the performance of OLES algorithm is still closest to standard LDA from human readability perspective.

5.2 Efficiency

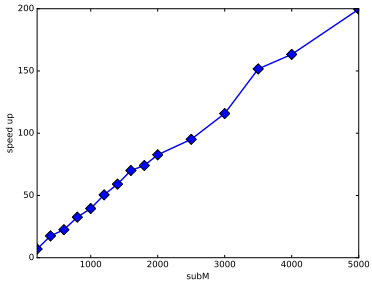
The most computationally intensive part of OLES algorithm is the clustering process of K points, which requires much less training time than LDA. In this section we demonstrate corresponding experiments for efficiency comparison.

We implement OLES algorithm in python/cython, and use gensim's LDA implementation[20] as benchmark. We test the OLES algorithm's efficiency promotion compared to standard LDA at different $subK$ and $subM$ on wikipedia dataset. Experiments are conducted on a Linux server with 2.1GHz Intel Xeon processor and 16GB RAM. Fig.5 shows the experimental results.

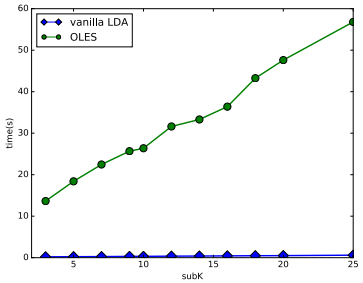
For a subset with moderate size (2,000 documents, 300,000 tokens, 7 topics) from a corpus with 300 pre-trained topics, OLES can construct the topic structures of subset in less than 200 milliseconds, which meets the requirement of online exploratory analysis.



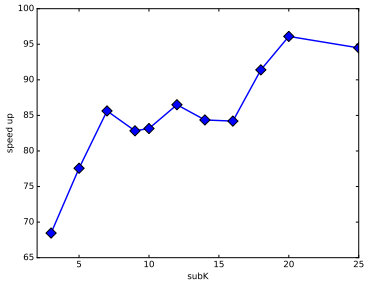
(a) Average time cost of 50 runs on wikipedia dataset, subK=7



(b) Efficiency speed up compared to standard LDA, subK=7



(c) Average time cost of 50 runs on wikipedia dataset, subM=2000



(d) Efficiency speed up compared to standard LDA, subK=7

Fig. 5: Efficiency promotion compared to standard LDA.

5.3 Scalability

With larger values of the number of corpus-level topics K, the number of documents in subset $subM$ and the vocabulary size T, the time needed for running an OLES algorithm also increases. In this case, We can further accelerate the algorithm with parallelization to meet the requirement of online processing.

We implement a parallelized version of OLES algorithm using MPI protocol, and run experiments with K=2000, T=50000, subM=5000 and subK=20. Fig.6 demonstrates the speed-up results, from which we can see the efficiency promotion is nearly linear to the number of processors.

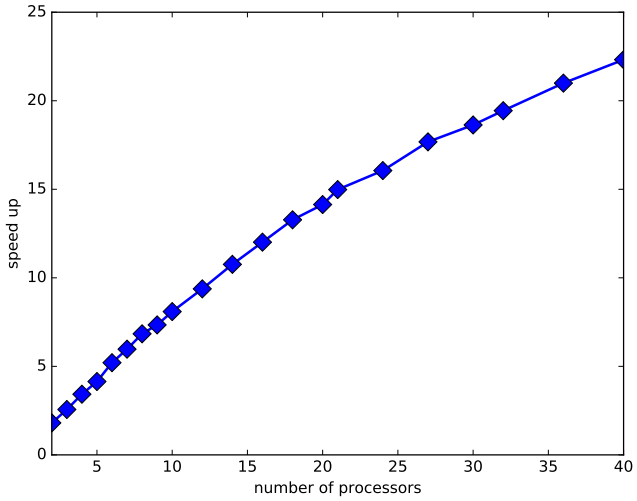
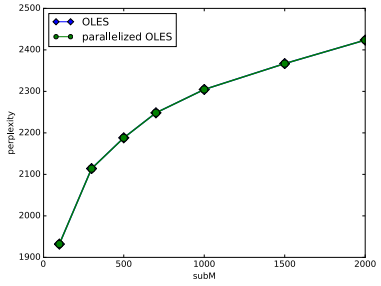
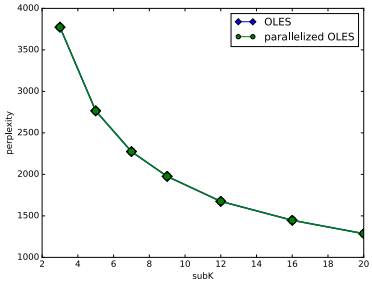


Fig. 6: Parallel speed-up results of subset modeling on wikipedia dataset, with K=2000, T=50000, subM=5000, and subK=20.

As the parallelization doesn't make any approximation, the perplexity results of parallelized OLES should be the same as serial version. For the sake of completeness, we give the comparison in Fig.7.



(a) Average perplexity of 50 runs on wikipedia dataset, with subK=7



(b) Average perplexity of 50 runs on wikipedia dataset, with subM=1000

Fig. 7: Perplexity comparison of parallelized OLES and serial OLES, the lines are almost completely overlapped.

Experimental results indicate that OLES algorithm can be parallelized with performance not declined.

6 Conclusion

In this work, we present OLES, an efficient algorithm for constructing topic structures in document subsets. The experimental results on Enron-Email and one-million wikipedia datasets show that the OLES algorithm outperforms all baseline methods in subset modeling, and obtains up to 200 times speed up compared with standard topic model. The OLES algorithm can be parallelized for further acceleration, the efficiency promotion is nearly linear to processor numbers in corresponding experiments.

References

1. 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)
2. Blei, D.M., Griffiths, T.L., Jordan, M.I.: The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. Journal of the ACM (JACM) 57(2), 7 (2010)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research 3(Jan), 993–1022 (2003)
4. Canini, K.R., Shi, L., Griffiths, T.L.: Online inference of topics with latent dirichlet allocation. In: AISTATS. vol. 9, pp. 65–72 (2009)
5. Cao, N., Sun, J., Lin, Y.R., Gotz, D., Liu, S., Qu, H.: Facetatlas: Multifaceted visualization for rich text corpora. IEEE transactions on visualization and computer graphics 16(6), 1172–1181 (2010)
6. Chaney, A.J.B., Blei, D.M.: Visualizing topic models. In: ICWSM (2012)
7. Chen, Y., Wang, L., Dong, M., Hua, J.: Exemplar-based visualization of large document corpus (infovis2009-1115). IEEE Transactions on Visualization and Computer Graphics 15(6), 1161–1168 (2009)
8. Dou, W., Yu, L., Wang, X., Ma, Z., Ribarsky, W.: Hierarchicaltopics: Visually exploring large text collections using topic hierarchies. IEEE Transactions on Visualization and Computer Graphics 19(12), 2002–2011 (2013)
9. Gardner, M.J., Lutes, J., Lund, J., Hansen, J., Walker, D., Ringger, E., Seppi, K.: The topic browser: An interactive tool for browsing topic models. In: NIPS Workshop on Challenges of Data Visualization. vol. 2 (2010)
10. Görg, C., Liu, Z., Kihm, J., Choo, J., Park, H., Stasko, J.: Combining computational analyses and interactive visualization for document exploration and sensemaking in jigsaw. IEEE Transactions on Visualization and Computer Graphics 19(10), 1646–1663 (2013)
11. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proceedings of the National academy of Sciences 101(suppl 1), 5228–5235 (2004)
12. Hoffman, M., Bach, F.R., Blei, D.M.: Online learning for latent dirichlet allocation. In: advances in neural information processing systems. pp. 856–864 (2010)
13. Hoffman, M.D., Blei, D.M., Wang, C., Paisley, J.W.: Stochastic variational inference. Journal of Machine Learning Research 14(1), 1303–1347 (2013)
14. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. pp. 50–57. ACM (1999)

15. Iwata, T., Yamada, T., Ueda, N.: Probabilistic latent semantic visualization: topic model for visualizing documents. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 363–371. ACM (2008)
16. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse processes 25(2-3), 259–284 (1998)
17. Li, W., McCallum, A.: Pachinko allocation: Dag-structured mixture models of topic correlations. In: Proceedings of the 23rd international conference on Machine learning. pp. 577–584. ACM (2006)
18. Mimno, D., McCallum, A.: Organizing the oca: learning faceted subjects from a library of digital books. In: Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries. pp. 376–385. ACM (2007)
19. Newman, D., Asuncion, A., Smyth, P., Welling, M.: Distributed algorithms for topic models. Journal of Machine Learning Research 10(Aug), 1801–1828 (2009)
20. Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45–50. ELRA, Valletta, Malta (May 2010), http://is.muni.cz/publication/884893/en
21. Smith, A., Hawes, T., Myers, M.: Hiérarchie: Interactive visualization for hierarchical topic models. In: Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces. pp. 71–78 (2014)
22. Yao, L., Mimno, D., McCallum, A.: Efficient methods for topic model inference on streaming document collections. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 937–946. ACM (2009)
23. Yuan, J., Gao, F., Ho, Q., Dai, W., Wei, J., Zheng, X., Xing, E.P., Liu, T.Y., Ma, W.Y.: Lightlda: Big topic models on modest computer clusters. In: Proceedings of the 24th International Conference on World Wide Web. pp. 1351–1361. ACM (2015)