Topic Exploration in Spatio-Temporal Document Collections

Kaiqi Zhao[1] Lisi Chen[2] Gao Cong[3]

Nanyang Technological University
{[1]kzhao002@e.,[2]lchen012@e.,[3]gaocong@}ntu.edu.sg

ABSTRACT

Huge amounts of data with both spatial and temporal information (e.g., geo-tagged tweets) are being generated, and are often used to share and spread personal updates, spontaneous ideas, and breaking news. We refer to such data as spatio-temporal documents. It is of great interest to explore topics in a collection of spatio-temporal documents.

In this paper, we study the problem of efficiently mining topics from spatio-temporal documents within a user specified bounded region and timespan, to provide users with insights about events, trends, and public concerns within the specified region and time period. We propose a novel algorithm that is able to efficiently combine two pre-trained topic models learnt from two document sets with a bounded error, based on which we develop an efficient approach to mining topics from a large number of spatio-temporal documents within a region and a timespan. Our experimental results show that our approach is able to improve the runtime by at least an order of magnitude compared with the baselines. Meanwhile, the effectiveness of our proposed method is close to the baselines.

1. INTRODUCTION

With the rapid development of online social media (e.g., Facebook, Flickr, Twitter, etc.) and GPS-enabled devices, huge amounts of data with both spatial and temporal information are being generated in an unprecedented scale. For example, Twitter, which allows users to compose tweets, has 320 million monthly active users who posted 500 million tweets per day, where 80% of the active users are on mobile[1]. Tweets are timestamped and they can be geo-tagged by enabling the geo-tagging functionality. Tweets are regarded as an up-to-date news source [6] and they have been analyzed for various types of human activity including execution of political actions, disaster management, crime prevention, emergency services, etc [17]. Such data can be regarded as a collection of spatio-temporal documents.

Since the spatio-temporal documents often contain information that is indicative of public views and interests [19, 12], it is of great interest to mine topics from the spatio-temporal document collections. Several studies of Online Analytical Processing (OLAP) aim at mining topics from text data (e.g., [32, 34]). Topic models, such as Probabilistic Latent Semantic Analysis (pLSA) [8], Latent Dirichlet Allocation (LDA) [4] and their variations, are proposed as one of the most effective text mining techniques and they have been successfully used for extracting topics from documents. In topic models, a document has a mixture of topics (e.g., "sport games", "entertainments", etc.) and each topic is modeled by a probabilistic distribution over a set of words (e.g., "football", "shoot", etc., for topic "sport games").

However, existing studies often neglect the fact that the location and created time of a document also play important roles in topic extraction. For example, a Twitter user in Boston may be interested in entirely different matters and news compared to a user in Hong Kong. Also, the trending topics in Boston will be different from those in U.S.A., as the trending topics in Boston may have more regional characteristics (e.g., "Boston Marathon"). Moreover, topics in June 2015 are substantially different from those in May 2015 (e.g., the emerging topic about the disease "MERS"). By finding topics in a region and a time period, many applications can be accomplished. For example, social scientists can find topics in different regions and time periods, and news providers can find breaking events from tweets within a region and a time period.

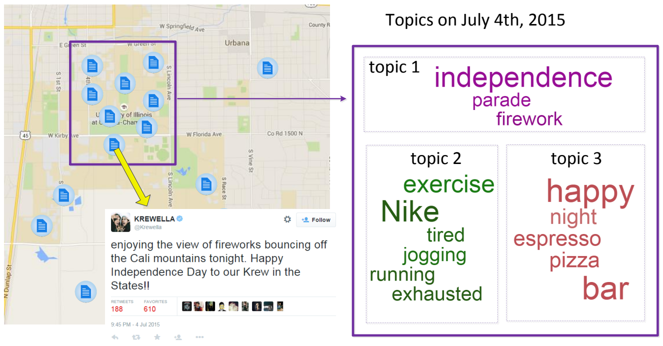
In this paper, we propose to mine topics from a collection of spatio-temporal documents within any user specified spatial region and time interval, which is to provide users with insights about events, trends, and public concerns within the specified region and time period.

Figure 1 exemplifies the problem of mining topics, where a user draws a bounded region (which is represented by the purple box on the map) and a time interval, July 4th, 2015. The blue icons indicate the spatio-temporal documents (e.g., tweets or news) containing text, location, and time information. The three topics extracted from the collection of spatial-temporal documents falling in the region on July 4th, 2015 are presented on the right. Since each topic in topic models is characterized by a probabilistic distribution over words, we use "Word Clouds" to visualize each topic. Word clouds use larger font to represent higher probability of a word in the topic. Note that for each topic we only show several words with the highest probabilities.

It is challenging to mine topics within a given region and a timespan efficiently and effectively. A straightforward method works as follows: Given a user specified region and timespan, we train a topic model on the documents falling in the region and the timespan using the existing techniques, such as LDA [4]. However, mining topics from spatio-temporal documents is very time-consuming.

[1]https://about.twitter.com/company (accessed date: Feb 9th, 2016)

SIGMOD'16, June 26-July 01, 2016, San Francisco, CA, USA
© 2016 ACM. ISBN 978-1-4503-3531-7/16/06...$15.00
DOI: http://dx.doi.org/10.1145/2882903.2882921



enjoying the view of fireworks bouncing off the Cali mountains tonight. Happy Independence Day to our Krew in the States!!

Figure 1: Topics within a Bounded Region and a Time Interval

The original problem of training topic models is NP-hard and collapsed Gibbs sampling is often used to approximately estimate the probabilistic distribution over words for each topic. However, the complexity of colllapsed Gibbs Sampling is still very high, i.e., $O(|W|KI)$, where K is the number of topics, I is the number of iterations of Gibbs sampling, and W is the set of word tokens in the document collection, which could be billions or trillions for a large document collection. For example, learning 100 topics our 3 month Twitter dataset in the New York city region, which contains 0.25 million tweets, took 8.31 minutes. As our Twitter data is a 1% sample of the geo-tagged tweet data, if we extend the result to a collection of all the 3-month geo-tagged tweets the New York city region based on the linear complexity, it would take 13.85 hours to learn 100 topics from the New York city region. Therefore, we need an efficient method to train topic models for each input of region and timespan.

To address the challenge, we propose the following new idea. We divide the collection of spatio-temporal documents into three-dimensional cells and train topic models for the documents within each cell. Then, to compute the topic models for a user specified region and timespan, we combine the topic models of the cells that are located in the specified region and timespan. To realize the idea, we propose the following novel techniques.

1. We propose a novel and efficient algorithm with a bounded error to combine two LDA topic models learnt from two document sets, both falling in the specified region and time period. Different from the existing topic modeling algorithms, our algorithm samples topics for sets of words instead of individual words. The complexity of training the LDA model on two document sets is $O((|W_1| + |W_2|)KI)$, where W_1 and W_2 are sets of word tokens in the two document sets. In contrast, our algorithm reduces the complexity to $O(K(K + |M_2| + |V_2|)I)$, where M_2 is the smaller document set and V_2 is its vocabulary. Their sizes are much smaller than W_1 and W_2.

2. We develop a new approach to partitioning the collection of spatio-temporal documents into an Octree [16], for indexing the documents. Instead of recursively partitioning the three dimensional space into equal-sized cells, we consider the word overlaps among the generated cells to determine how to perform the partitioning. Our partitioning mechanism enables our combining algorithm to achieve better accuracy, although our proposed combining algorithm works for any kind of partitioning. We also develop a mechanism to determine on which cells we pre-train topic models to guarantee a given accuracy threshold.

3. We develop a framework to find the cells whose regions and time periods fall in or overlap with the user specified region and time period. Then we employ our proposed algorithms to combine the pre-trained topic models in these cells to obtain the topics for documents in the specified region and time period.

In summary, the contributions in this study are threefold.

First, we define a practical and novel problem of extracting topics for exploration from spatio-temporal documents within a region and a timespan.

Second, we propose a novel approach comprising the aforementioned key techniques that is capable of efficiently mining topics from a large number of spatio-temporal documents within a given region and timespan.

Third, we conduct an extensive experimental study for evaluating our proposals on three large real-world datasets collected from Twitter, Meetup, and Wikipedia, respectively. Our experiments show that our proposed topic mining framework is able to improve on the runtime by at least an order of magnitude compared to LDA [4] and a variation of LDA [2] that supports online updates (online-LDA). Meanwhile, the effectiveness of the proposed topic extraction framework, measured by perplexity, is close to those of LDA and online-LDA. We also use some example topics extracted by our framework to show that the quality of the extracted topics is comparable to those of LDA and online-LDA.

The rest of this paper is organized as follows. Section 2 introduces the preliminaries and the related work. Section 3 defines the problem of mining topics given a bounded region and a timespan. Section 4 presents an overview of the proposed solution. Section 5 introduces our methods. We present the experimental studies in Section 6. Section 7 concludes the study.

2. PRELIMINARIES & RELATED WORK

Table 1: Summary of Notations

Notation	Meaning
d	a spatio-temporal document $\langle id, l, t_c, w\rangle$
$z^{(m,n)}$	topic assigned to the n-th term of the m-th document
$p(v\|k)$	probability of appearance of word v given topic k
$c^k_{*,v}$	the count that topic k is assigned to word v
$c^k_{m,*}$	the count that topic k is assigned to the m-th document
α, β	hyper parameters of LDA
R	input spatial region
$[t_b, t_e]$	input timespan

Table 1 presents the frequently used notations.

Preliminary on Topic Models. Topic models are probabilistic models for discovering "topics" in a collection of documents. Each document may contain several topics, e.g., "sports", "digital devices", etc. If a document is more likely to be a "sports" document, words such as "football", "exercise", etc., would be more likely to appear in the document than "firework" or "pizza", etc. Topic models aim to reveal such latent semantic structure of a collection of documents. In topic models, a document is modeled as a mixture of topics and each topic is represented by a multinomial distribution over words.

Latent Dirichlet Allocation (LDA) [4] is arguably the most successful topic model to date. LDA is a generative model which generates topic distribution θ_d for each document d from a Dirichlet distribution $Dir(\alpha)$ and the word distribution ϕ for each topic from another Dirichlet distribution $Dir(\beta)$. The parameters α and β are hyper parameters of LDA which are used for smoothing. For each word token v in each document d, LDA draws a topic k from θ_d and then generates v from the corresponding word distribution ϕ_k.

Estimating the parameters, i.e., word distribution for topics ϕ, in LDA is intractable, and several algorithms (e.g., variational Expectation-Maximization algorithm, collapsed Gibbs sampling, etc.) have been proposed to solve the problem. Collapsed Gibbs sampling is a commonly used method to estimate the parameters for LDA. It works as follows: 1) Randomly assign a topic (from a given number of topics K) to each word token in the document; 2) Iteratively update the topic assignment for each word token according to Eq. (1).

$$p(z^{(m,n)} = k|Z^{-(m,n)}, \alpha, \beta) \propto (c_{m,*}^{k,-(m,n)} + \alpha) \times \frac{c_{*,v}^{k,-(m,n)} + \beta}{\sum_r c_{*,r}^{k,-(m,n)} + |V|\beta}, \quad (1)$$

where $z^{(m,n)}$ denotes the topic assignment for the n-th word in document m, $Z^{-(m,n)}$ denotes the topic assignment for word tokens excluding the n-th word token in document m, $c_{m,v}^{k,-(m,n)}$ is the count of assigning topic k to word v in document m excluding the topic assignment of the n-th word token in document m, and wildcard * stands for any document/word. The set V is the set of unique words, which is also called vocabulary, in the collection of documents. Intuitively, the first part of the equation tells how prevalent is topic k in document m, while the second part tells how prevalent is word v belonging to topic k across all documents. Since Gibbs sampling updates the topic assignment for each word token (m, n) by computing Eq. (1) for K topics in each iteration, the complexity of the algorithm is $O(|W|KI)$, where W is the set of word tokens in the collection of documents, and I is the number of iterations. It could be very slow when the collection of documents is large. After Gibbs sampling, the language model of a topic (posterior word distribution given a topic) $p(v|k)$ can be estimated as:

$$p(v|k) = \frac{c_{*,v}^{k} + \beta}{\sum_{r \in V} c_{*,r}^{k} + |V|\beta}, \quad (2)$$

where $c_{*,v}^{k}$ is the count of assigning topic k to word v in all documents.

Several studies exist on parallel implementation of LDA [26, 22, 1]. The main problem in the parallel LDA systems is that the topic-word count $c_{*,v}^{k}$ in Eq. (1) is shared by all processors. To make the topic-word counts consistent, all machines need to communicate with each other at the end of each iteration of Gibbs sampling, which incurs expensive communication cost. These systems can at most achieve a sub-linear speedup and training LDA models remains expensive.

Based on LDA, several online learning algorithms are proposed for text streams [2, 28] to speed up the learning process. They aim to sample topic assignments for new documents using a topic model learnt from historical data and combine the topic assignments for new documents to construct a new topic model. Sumait et al. [2] propose an online updating method (Online-LDA) to sample new documents using the counts of topic assignments in the historical data ($c_{old,*,v}^{k}$) as a part of the prior distribution, i.e., we set $\beta' = c_{old,*,v}^{k} + \beta$. Let $c_{new,m,v}^{k,-(m,n)}$ be the count of assigning topic k to word v in the m-th new document, excluding the topic assignment for the n-th word token in the m-th new document. The update function for online-LDA is:

$$p(z_{new}^{(m,n)} = k|Z_{new}^{-(m,n)}, Z_{old}, \alpha, \beta) \propto (c_{new,m,*}^{k,-(m,n)} + \alpha) \times \frac{c_{new,*,v}^{k,-(m,n)} + c_{old,*,v}^{k} + \beta}{\sum_r c_{new,*,r}^{k,-(m,n)} + c_{old,*,r}^{k} + |V|\beta}, \quad (3)$$

where $Z_{new}^{-(m,n)}$ is the topic assignments to word tokens in the new document collection excluding the n-th word in document m, and Z_{old} is the topic assignments to word tokens in historical data.

Online-LDA has the same complexity as LDA on the set of new documents, i.e., $O(|W_{new}|KI)$, where W_{new} is the set of word tokens in the new documents. But online-LDA saves the computation for sampling topics on the old documents. Online-LDA is the state-of-the-art technique which can be used to learn topics in a given region and timespan. However, to the best of our knowledge, no existing work proposes techniques for combining two LDA topic models pre-trained from two document sets into one model as we do in this work.

Data Cube and Text Cube. Our work is inspired by the concept of data cube [7], which is widely used for Online Analytical Processing (OLAP) on multi-dimensional data with operations like slice, dice, roll-up, and drill-down. Each row in the multi-dimensional text database contains some structured dimensions (i.e., attributes) and unstructured text data. The existing work on analyzing and exploring multi-dimensional text database is closest to our problem.

Lin et al. [13] propose a text-cube model on multi-dimensional text database. Text cube is a data cube where each cell aggregates a set of documents with matching attribute values in a subset of dimensions. Based on the text cube, they study the problem of processing an IR query (i.e., a set of terms) with constraints on dimensions to retrieve relevant documents. Simitsis et al. [20] propose a keyword driven OLAP system over a collection of multidimensional text data. Operations of roll-up and drill-down can be performed efficiently and accurately based on the content and the link-structure of a dynamically selected document subset during the query time. In addition, Ding et al. [5] study the problem of finding top-k most relevant cells in text cube for a keyword query. Similarly, based on the text cube, Zhao et al. [35] develop the TEXplorer system to rank candidate dimensions and cells for a keyword query.

However, those text-cube based proposals do not consider the semantic meaning (i.e., topics) of each term in documents. They are actually orthogonal with our work, and our proposed techniques can be integrated into these proposals for exploring spatial-temporal topics of data cube. This is actually a great motivation for our work. Moreover, our proposed combining algorithm in this paper fits perfectly to support the roll-up operation for topics of a text cube.

Zhang et al. [32] develop a topic-cube model for a multidimensional text database. Specifically, the topic cube can be viewed as a standard data cube augmented by a topic dimension based on a hierarchical topic tree and the objective of the topic-cube is to support users to drill-down and roll-up the text dimension along the topic hierarchy. The topic tree in [32] is based on the topic model generated from all documents in the dataset. However, our problem requires the topic model to be generated from the documents in the user specified spatial region and time interval. Moreover, topic-cube does not support mining topics for user specified regions. Therefore, topic cube cannot be applied to our problem.

Content Exploration. Some recent work aims at developing online data exploration mechanism based on some clustering methods. Zhang et al. [33] propose MiTexCube by augmenting each cell with micro-clusters that make the online processing more efficient. More recently, Feng et al. [6] propose a system called StreamCube to explore events over the spatio-temporal Twitter stream by clustering hashtags. Each hashtag in StreamCube is represented by a vector of terms/hashtags that co-occurred with the hashtag. Consequently, the clusters are generated based on the tweets that have hashtag(s), which only take 11% tweets according to a statistical study [10]. One can use keyword extracted from the text instead of hashtags to address the sparsity problem in StreamCube. However, we focus on the problem that efficiently mines topics of a collection of documents in the user specified region and timespan, as well as the topic assignment of each selected document. The comparison of clustering and topic modeling techniques is not our focus and we leave this as future work.

Angel et al. [3] propose the Grapevine system that aims to track entities in a large scale of documents. Grapevine extracts real-word entities (e.g., Barack Obama) from a collection of text data (e.g., blog posts, news articles, and tweets) using available entity extraction algorithms (e.g., [18]). Given a temporal and demographic restriction (e.g., location, gender), the topic in Grapevine is defined by the most talked-about entities for the chosen dimensions. However, our topics are discovered from topic models where each topic is represented by a probabilistic distribution over words. In addition, Grapevine uses semantic locations (country, state, city), while we use coordinate locations.

Exploring Spatio-Temporal Information in Documents. There exists a host of work on extracting spatial or/and temporal information from text data (e.g., [23, 11]). Several studies aim at mining geographical topics from a collection of spatio-temporal documents (e.g., [30, 9, 21, 31, 36]), and they mine both geographical regions, each represented by a Gaussian distribution, and topics for the whole collection. In some proposals [2, 27], the topic models in different time slices are compared to detect burst/emerging topics. Mathioudakis et al. [15] propose an approach to monitoring trends over the Twitter stream by detecting and grouping bursty keywords from tweets. Yin et al. [29] propose a user behavior model that combines the influences of topics related to users' intrinsic interests and the topics related to temporal context. Moreover, Strötgen et al. [24] try to discover event-location pairs by considering the co-occurrence of geographical and temporal expressions in document content. However, these problems substantially differ from our problem.

3. PROBLEM STATEMENT

We introduce our data model and define the problem of mining topics over a specified region and timespan.

3.1 Data Model

Definition 1: Spatio-Temporal Document. A spatio-temporal document is denoted by a quadruple $d = \langle id, l, t_c, w\rangle$, where id is the document id, which is assigned based on t_c, the creation time of d, l is a location with latitude and longitude, and w is a sequence of word tokens from the vocabulary $V = \{v_1, v_2, ..., v_{|V|}\}$. We use $d.w[i]$ to denote the i-th token in $d.w$. □

The spatio-temporal documents in Definition 1 can be geo-tagged tweets in Twitter, geo-tagged news in news portals, geo-tagged photos with tags in Flickr, Points of Interest (POIs) and check-ins with text descriptions in Foursquare, etc. Without loss of generality, we consider each document has only one location. When a document contains more than one location, we duplicate the documents for each location. For example, if a tweet about European SuperCup is related to three locations, (e.g., Barcelona, Sevilla, and Tblisi), we generate three copies of the tweet and each copy is assigned to one location.

3.2 Problem Definition

Based on the definition of topic in Section 2, we present our problem statement.

Definition 2: Topic Mining over Documents within a Specified Region and a Timespan Query (TMRT Query). Let D be a set of spatio-temporal documents, R be a rectangular region, and $[t_b, t_e]$ be a time period. TMRT query is to mine K topics from documents in D whose locations fall in R and creation time falls in $[t_b, t_e]$, as well as the topic assignment of each selected document, based on a given topic model. □

In this paper, we use LDA to mine topics for spatio-temporal documents because it is one of the most commonly used topic models.

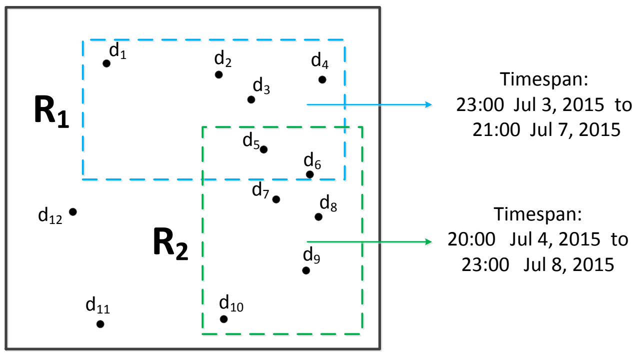


Figure 2: Example

Table 2: Information of Documents

doc	time of creation	tokens
d_1	09:42 Jul 3, 2015	yummy, pizza
d_2	10:18 Jul 6, 2015	espresso, bar, night
d_3	13:43 Jul 4, 2015	Barcelona, champion, excited
d_4	19:22 Jul 7, 2015	football, excited
d_5	17:40 Jul 6, 2015	iPhone, Nike, exercise
d_6	12:23 Jul 8, 2015	sleep, tired, holiday
d_7	22:57 Jul 3, 2015	Barcelona, sightseeing
d_8	18:41 Jul 4, 2015	marathon, exhausted
d_9	21:42 Jul 4, 2015	holiday, parade, independence
d_{10}	23:33 Jul 4, 2015	independence, firework
d_{11}	17:19 Jul 3, 2015	3D, movie, boring
d_{12}	10:40 Jul 7, 2015	sigmod, deadline

Table 3: Final Topic Assignment of R_1

document	topic of tokens
d_2	espresso (k_0), bar (k_1), night (k_1)
d_3	Barcelona (k_2), champion (k_2), excited (k_0)
d_4	football (k_2), excited (k_0)
d_5	iPhone (k_1), Nike (k_2), exercise (k_2)

Example 1: Let $D = \{d_1, d_2, ..., d_{12}\}$ be a set of spatio-temporal documents. The location of each document is presented in Figure 2. The creation time and the word tokens of each document are presented in Table 2. The two rectangles R_1 and R_2 and their corresponding timespans in Figure 2 represent two TMRT queries.

Table 4: Final Topic Assignment R_2

document	topic of tokens
d_5	iPhone (k_0), Nike (k_1) exercise (k_1)
d_6	sleep (k_0), tired (k_1), holiday (k_0)
d_9	holiday (k_0), parade (k_2), independence (k_2)
d_{10}	independence (k_2), firework (k_2)

Based on the set of documents falling in the corresponding spatial regions and timespans of R_1 and R_2, which are presented in the first column of Table 3 and Table 4, respectively, we mine LDA topic models by employing Gibbs sampling algorithm. Suppose the number of topics (i.e., "K") is set to 3, the final topic assignments for each document in R_1 and R_2 are presented in the second column of Table 3 and Table 4, respectively. The three topics of R_1 are given as follows (suppose the prior of word distribution of each topic $\beta = 0$ in LDA):

k_0: <p(excited|k_0)=0.67, p(espresso|k_0)=0.33>
k_1: <p(bar|k_1)=0.33, p(night|k_1)=0.33,
p(iPhone|k_1)=0.33>
k_2: <p(Barcelona|k_2)=0.2, p(champion|k_2)=0.2,
p(football|k_2)=0.2, p(Nike|k_2)=0.2,
p(exercise|k_2)=0.2>.

The topics of R_2 is as follows:
k_0: <p(holiday|k_0)=0.5, p(iPhone|k_0)=0.25,
p(sleep|k_0)=0.25>
k_1: <p(Nike|k_1)=0.33, p(exercise|k_1)=0.33,
p(tired|k_1)=0.33>
k_2: <p(independence|k_2)=0.5, p(parade|k_2)=0.25,
p(firework|k_2)=0.25>.

□

4. FRAMEWORK OVERVIEW

Based on LDA, we aim at mining K topics from a spatio-temporal document collection D for a TMRT query Q, which comprises a user specified region R and time interval [t_b, t_e]. In general, the collection of spatio-temporal documents is treated as a data warehouse and we conduct topic mining over a subset of documents falling in the region and time interval specified in Q. A straightforward solution would work as follows: We first find a set of documents D' such that for all $d \in D'$, d locates in R and $d.t_c \in [t_b, t_e]$. Then we learn K topics from the documents in D' using the collapsed Gibbs sampling technique [14]. However, as discussed in Section 1, it is time-consuming ($O(|W|KI)$) to train the topic model every time a TMRT query is submitted. Hence, we need a more efficient mechanism to handle the TMRT query.

An underlying idea of many approaches to performing OLAP is to organize the data into data cubes with dimensional hierarchies and summary statistics (e.g., [6, 32, 13]). With the help of data cubes, we are able to efficiently explore the data by different granularity levels [13]. Inspired by the idea of data cube, we organize the spatio-temporal documents into cells in a hierarchical index, and pre-compute topic models for a set of selected cells.

The remaining challenges of processing the TMRT query include: (1) How to leverage the pre-trained models associated with indexed cells to efficiently compute the topic model for the given query Q with region R and timespan $[t_b, t_e]$; (2) How to organize the documents into a hierarchical index, and which cells in the hierarchical index do we pre-compute topic models to help accelerate online topic mining for the TMRT query?

To address the first challenge, we propose a novel idea for efficiently combining the topic models of two cells (Section 5.1). Our idea is to accelerate the topic learning process by sampling topics for a set of word tokens instead of sampling topics for individual word tokens. We prove that our combining algorithm has a bounded error compared with the online-LDA algorithm. Based on our idea of combining topic models, we propose an online topic mining algorithm to search for cells which are completely or highly covered by the spatial region and timespan of query Q, and combine the models of these cells using our combining algorithm (Section 5.4).

To address the second challenge, we propose to organize the spatio-temporal documents based on the idea of Octree [16]. Octree is an extension of Quadtree in a three-dimensional space. It recursively partitions the space into eight equal-sized cells. Different from the traditional Octree in which a cell is partitioned into eight equal-sized subcells, we do the partitioning differently – we consider two objectives in determining how to partition a cell: 1) minimizing the word overlap among the subcells; and 2) balancing the number of documents among the subcells (Section 5.3). In addition, we propose an algorithm to determine on which cells to pre-train topic models and when to stop partitioning such that the space requirement does not exceed a specified threshold and the error of combining topic models does not exceed a predefined bound.

Figure 3 illustrates the framework of our approach. The documents are indexed by an Octree. Given a TMRT query with a rectangle and a timespan, we traverse the Octree to retrieve the topic models of the cells covered by the rectangle and the timespan. Next we combine these retrieved topic models. Finally we output the K topics of the new topic model computed by our combining algorithm.

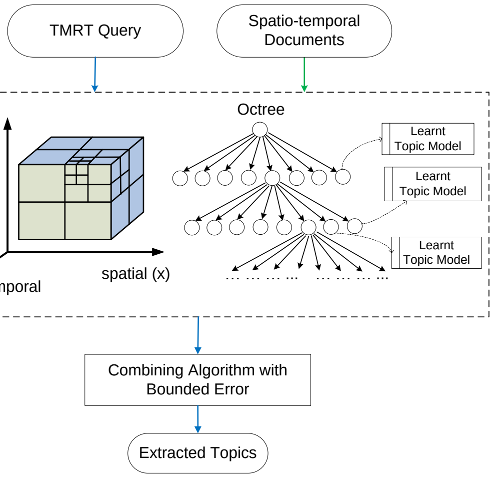


Figure 3: Framework

5. ALGORITHMS

In this section, we present the details of our algorithms for handling the TMRT queries. Specifically, we first present our sampling algorithm for combining the topic models of two Octree cells and its extension to combining multiple cells in Section 5.1. Then, we present a method to handle the cells that overlap with, but are not contained by the user specified region and timespan (Section 5.2). We present our method for partitioning the spatio-temporal documents into an Octree and the offline pre-computation for Octree cells in Section 5.3. Finally, we give our online topic learning algorithm for a TMRT query in Section 5.4.

5.1 Combining Topics of Two Octree Cells

In this subsection, we consider the case that the given region R and time interval $[t_b, t_e]$ in the TMRT problem exactly cover two Octree cells, for which we have pre-trained topic models. We will consider the general case and how to construct the Octrees in the subsequent subsections. We aim to combine the two pre-trained topic models into one topic model within R and $[t_b, t_e]$. The combining is challenging in that the topics in one model cannot be simply associated to the other model. Example 2 shows the pre-trained topics of two cells, which are represented by distribution over words. Even though k_2 in cell 1 is similar to k_1 in cell 2 (about sports), they are not exactly the same topic because their distributions are different. Moreover, a topic in one cell could be similar to multiple topics in the other cell. For example, k_0 in cell 2 is similar to k_0 and k_1 in cell 1. Due to the incompatibility of topics between the two models, we cannot simply combine them. We need to re-sample the topic models of the cells to make them compatible for combination.

Example 2: Pre-trained topics for cell 1:
k_0: <p(`iPhone`|k_0)=0.43, p(`Android`|k_0)=0.57>
k_1: <p(`bar`|k_1)=0.33, p(`night`|k_1)=0.33, p(`iPhone`|k_1)=0.33>
k_2: <p(`Barcelona`|k_2)=0.25, p(`champion`|k_2)=0.25, p(`football`|k_2)=0.25, p(`Nike`|k_2)=0.25>
Pre-trained topics for cell 2:
k_0: <p(`holiday`|k_0)=0.5, p(`iPhone`|k_0)=0.25, p(`sleep`|k_0)=0.25>
k_1: <p(`Nike`|k_1)=0.33, p(`exercise`|k_1)=0.33, p(`tired`|k_1)=0.33>
k_2: <p(`independence`|k_2)=0.5, p(`parade`|k_2)=0.25, p(`firework`|k_2)=0.25>.
□

There exist two solutions to learn the topic model for a TMRT query: 1) re-training an LDA model [4] for the spatio-temporal documents in the two cells; and 2) re-sampling the topics for one of the two Octree cells given the topic assignment of the other one using online-LDA [2]. However, both solutions suffer from high computation cost. Let the sets of word tokens in the two cells be W_1 and W_2, respectively, where W_2 is for the cell containing fewer words, i.e., $|W_2| < |W_1|$. The first solution needs to sample topics for all the documents in the two cells, and its complexity is $O((|W_1| + |W_2|)KI)$. The second solution only samples topics for documents in the cell containing fewer word tokens, and thus the training time can be reduced at least by half, i.e., $O(|W_2|KI)$. Although the second solution utilizes the pre-trained model of one cell, it fails to utilize the other pre-trained model.

To this end, we propose a combining algorithm that is able to leverage the pre-trained models of both cells. We reduce the complexity to $O((K + |M_2| + |V_2|)KI)$, where M_2 and V_2 are the set of documents and the vocabulary of the smaller cell, respectively. Note that $K + |M_2| + |V_2|$ is much smaller than $|W_2|$. We also prove that our algorithm has a bounded sampling error compared to online-LDA.

Similar to online-LDA, we use the topic assignment of one cell to re-train the model for the other cell. The significant difference is that we do not sample the topic for each word token in the documents one by one as in online-LDA. Instead, we sample the topic for word tokens that are assigned to the same topic in the pre-trained model simultaneously. Since this treatment is an approximation to online-LDA, we prove an error bound of our sampling algorithm compared to online-LDA. To the best of our knowledge, the idea is new although the topic model has attracted extensive research attention.

Let Z_1 and Z_2 be the topic assignment for the two sets of word tokens W_1 and W_2, respectively. Without loss of generality, we assume that $|W_1| > |W_2|$. We want to use the topic assignments Z_1 of the pre-trained model for cell 1 to resample Z_2. Consider the fact that the topic assignments of the pre-trained model divide the set of tokens W_2 into K subsets: $\{W_{21}, W_{22}, ..., W_{2K}\}$. Thus, we can jointly sample the word tokens of each subset according to the joint posterior distribution given the other topic assignments: $p(Z_2^{W_{2i}} = k | Z_2^{-W_{2i}}, Z_1, \alpha, \beta)$. Here, we denote $Z_2^{W_{2i}} = k$ as the event that all word tokens in W_{2i} are assigned to topic k and $Z_2^{-W_{2i}}$ as the topic assignments for the other subsets. The parameter α is prior of topic distribution and β is prior of word distribution for each topic as used in LDA.

Inference: We use collapsed Gibbs sampling to infer topics for the word tokens. Specifically, we compute the complete likelihood of the model, i.e., $p(W_2, Z_2, \Theta, \Phi | Z_1, \alpha, \beta)$, where Θ is the topic distribution and Φ is word distributions for topics. To compute the posterior distribution of assigning a topic to a set of word tokens (i.e., $p(Z_2^{W_{2i}} = k | Z_2^{-W_{2i}}, Z_1, \alpha, \beta)$), we integrate Θ and Φ out and compute the posterior as:

$$p(Z_2^{W_{2i}} = k | Z_2^{-W_{2i}}, Z_1, \alpha, \beta)$$
$$\propto \prod_{d_m \in D_{2i}} (c_{2,m,*}^{k,-W_{2i}} + \alpha)^{c_{2,m,*}^{i}} \prod_{v \in V_{2i}} \left(\frac{c_{2,*,v}^{k,-W_{2i}} + \beta'_{kv}}{\sum_r c_{2,*,r}^{k,-W_{2i}} + \beta'_{kr}}\right)^{c_{2,*,v}^{i}}, \tag{4}$$

where D_{2i} is the set of documents that contain the word tokens in W_{2i} and V_{2i} is the set of unique words in W_{2i}. The pseudo count β' is the sum of actual counts in cell 1 plus β, i.e., $\beta'_{kv} = c_{1,*,v}^{k} + \beta$, where $c_{1,*,v}^{k}$ is the count of assigning topic k to word v in cell 1. The count $c_{2,m,*}^{i}$ is the number of words in document d_m that are also in W_{2i}. Similarly, $c_{2,*,v}^{i}$ is the number of occurrences of word v in W_{2i}. The first part of Eq. (4) is the likelihood of assigning topic k to the current set of documents (D_{2i}) without considering the topic assignments on W_{2i}. The second part of Eq. (4) is the likelihood of assigning topic k to the unique words in W_{2i} by excluding the topic assignments of W_{2i}. The two parts of Eq. (4) correspond to the two parts in LDA (Eq. (1)).

Combining: To combine the topic model of two cells, we apply Eq. (4) to sample a topic for each word token set. We repeat the sampling process until the complete likelihood converges or the sampling process exceeds a large number of iterations, i.e., 1000. Let $\widetilde{c}_{2,*,v}^{k}$ be the final count of assigning topic k to word v. We combine the topic model by summing up the actual counts of the two cells: $\widetilde{c}_{*,v}^{k} = c_{1,*,v}^{k} + \widetilde{c}_{2,*,v}^{k}$. Therefore, the word distribution of each topic k for the combined model ϕ_{kv} could be estimated as:

$$\widetilde{\phi}_{kv} = \frac{\widetilde{c}_{*,v}^{k} + \beta}{\sum_r \widetilde{c}_{*,r}^{k} + |V|\beta}. \tag{5}$$

Complexity: The computation of Eq. (4) has time complexity $O(|M_{2i}| + |V_{2i}|)$ because we need to compute the production over the documents and words. We have K token sets in total and we need to compute Eq. (4) for K topics to sample the topic for each token set. Therefore, the total time complexity is $O(K^2(|M_{2i}| + |V_{2i}|)I) = O((K + |M_2| + |V_2|)KI)$, where M_2 is the set of documents and V_2 is the set of unique words in cell 2. Since $K + |M_2| + |V_2|$ is much smaller than $|W_2|$ (e.g., less than 10% of $|W_2|$ for tweets and even lower for long documents).

Sampling Error: We next prove that our algorithm has a bounded error compared to the topic models returned by online-LDA.

THEOREM 1. *Let $\phi_k^{(1,2)}$ and $\widetilde{\phi}_k^{(1,2)}$ be the language models for topic k after we combine cell 2 to cell 1 using online-LDA (Eq. (3)) and our approach (Eq. (4)), respectively. Let W_1 and W_2 be the word token sets of cell 1 and cell 2, respectively. Then, the expectation of Euclidean distance between two models $\mathbb{E}d(\phi_k^{(1,2)}, \widetilde{\phi}_k^{(1,2)}) < \sqrt{2}\frac{|W_1 \cap W_2|+|W_2|-1}{|W_1|+|W_2|}$.*

PROOF. Let $C_2^k = \sum_r c_{2,*,r}^k$, $\widetilde{C}_2^k = \sum_r \widetilde{c}_{2,*,r}^k$ and $C_1^k = \sum_r c_{1,*,r}^k$. Since the two sampling algorithms are based on the same prior α and $\beta'_{kv} = c_{1,*,v}^k + \beta$, we can assume $C_2^k \approx \widetilde{C}_2^k$. This assumption means the two algorithms will results in a similar topic proportion, i.e., the total counts for each topic is similar. Then we infer the bound as:

$$\begin{aligned}\mathbb{E}d(\phi_k^{(1,2)}, \widetilde{\phi}_k^{(1,2)})^2 &\approx \mathbb{E}\sum_{r \in W_1 \cap W_2}\left(\frac{c_{2,*,r}^k - \widetilde{c}_{2,*,r}^k}{C_2^k + C_1^k + |V|\beta}\right)^2 \\ &\leq \mathbb{E}\sum_{r \in W_1 \cap W_2}\frac{(c_{2,*,r}^k)^2 + (\widetilde{c}_{2,*,r}^k)^2}{(C_2^k + C_1^k + |V|\beta)^2} \\ &\leq \frac{\mathbb{E}(\sum_{r\in W_1\cap W_2} c_{2,*,r}^k)^2 + \mathbb{E}(\sum_{r\in W_1\cap W_2} \widetilde{c}_{2,*,r}^k)^2}{(\frac{|W_1|+|W_2|}{|W_2|}C_2^k + |V|\beta)^2}.\end{aligned} \tag{6}$$

We next apply the definition of variance $\mathbb{E}(X-\bar{X})^2 = \mathbb{E}(X^2) - (\mathbb{E}X)^2$ to the inequality. Let $X = \sum_{r\in W_1\cap W_2} c_{2,*,r}^k$, then we have:

$$\begin{aligned}\mathbb{E}(\sum_{r\in W_1\cap W_2} c_{2,*,r}^k)^2 &= (\frac{|W_1 \cap W_2|}{|W_2|}C_2^k)^2 \\ &+ \mathbb{E}(\sum_{r\in W_1\cap W_2} c_{2,*,r}^k - \frac{|W_1 \cap W_2|}{|W_2|}C_2^k)^2 \\ &< (\frac{|W_1 \cap W_2|}{|W_2|}C_2^k)^2 + \sum_{r\in W_1\cap W_2}\mathbb{E}(c_{2,*,r}^k - \frac{C_2^k}{|W_2|})^2.\end{aligned} \tag{7}$$

Since the maximum value of the second term appears when we assign all words to one topic, i.e., it is not hard to infer that the second term is smaller than $\frac{|W_2|-1}{|W_2|}(C_2^k)^2$. We apply similar inference on the term $\mathbb{E}(\sum_{r\in W_1\cap W_2} \widetilde{c}_{2,*,r}^k)^2$ in Eq. (6) to obtain the error bound of combining two topic models: $\sqrt{2}\frac{|W_1\cap W_2|+|W_2|-1}{|W_1|+|W_2|}$. □

According to Theorem 1, the smaller overlap of words between two cells will yield a lower error bound. Combining small cells to large cells yield a lower error bound.

Combining Multiple Cells: We extend the aforementioned method for combining two Octree cells to multiple cells. Suppose that the given region R and time interval $[t_b, t_e]$ exactly cover n cells, each having a pre-trained topic model. We can apply our combining method $n-1$ times on the n cells to learn the topic model. Next, we show that the error bound of combining multiple cells is parameterized by the number of combinings.

THEOREM 2. *Let $\phi_k^{(1,\ldots,n)}$ and $\widetilde{\phi}_k^{(1,\ldots,n)}$ be the language models for topic k obtained by combining topics of n cells according to Eq. (3) and Eq. (4), respectively. Let $W_1, \ldots, W_n$ be the sets of word tokens of n cells, respectively. Then, the expectation of Euclidean distance between two models $\mathbb{E}d(\phi_k^{(1,\ldots,n)}, \widetilde{\phi}_k^{(1,\ldots,n)}) < f(n) = \sqrt{2}\,\frac{|\cup_{i=2}^n W_i \cap W_1| + \sum_{i=2}^n |W_i| - 1}{\sum_{i=1}^n |W_i|}$.*

PROOF. Let $C_i^k = \sum_r c_{i,*,r}^k$, $\widetilde{C}_i^k = \sum_r \widetilde{c}_{i,*,r}^k$, where $1 \leq i \leq n$. Similar to Theorem 1, we assume $C_i^k \approx \widetilde{C}_i^k$. Then we have:

$$\begin{aligned}\mathbb{E}d(\phi_k^{(1,\ldots,n)}, \widetilde{\phi}_k^{(1,\ldots,n)})^2 &\approx \mathbb{E}\sum_{r\in\cup_{i=2}^n W_i\cap W_1}\left(\frac{\sum_{i=2}^n c_{i,*,r}^k - \widetilde{c}_{i,*,r}^k}{\sum_{i=1}^n C_i^k + |V|\beta}\right)^2 \\ &\leq \mathbb{E}\sum_{r\in\cup_{i=2}^n W_i\cap W_1}\frac{(\sum_{i=2}^n c_{i,*,r}^k)^2 + (\sum_{i=2}^n \widetilde{c}_{i,*,r}^k)^2}{(\sum_{i=1}^n C_i^k + |V|\beta)^2} \\ &< \frac{2(\frac{|\cup_{i=2}^n W_i\cap W_1| + \sum_{i=2}^n |W_i| - 1}{\sum_{i=2}^n |W_i|}\sum_{i=2}^n C_i^k)^2}{(\frac{\sum_{i=1}^n |W_i|}{\sum_{i=2}^n |W_i|}\sum_{i=2}^n C_i^k + |V|\beta)^2}.\end{aligned} \tag{8}$$

Thus, the error bound is $\sqrt{2}\,\frac{|\cup_{i=2}^n W_i\cap W_1| + \sum_{i=2}^n |W_i| - 1}{\sum_{i=1}^n |W_i|}$. □

5.2 Processing Partially Covered Cells

Since the given query with spatial boundary R and time interval $[t_b, t_e]$ may not cover an Octree cell completely, but only a part of it, we need to consider how to utilize the partial covered cells. Suppose the query completely covers n cells and partially covers n' cells and let $\widetilde{\phi}^{(n)}$ be the language model after combining the n completely covered cells. One solution is to apply online-LDA to the documents covered by the region and timespan of the given TMRT query in the n' partially covered cells, using $\widetilde{\phi}^{(n)}$ as the prior of language models. However, when the number of documents in the n' partially covered cells is large, it is not efficient to apply online-LDA to these n' partially covered cells.

Notice that for the cell that has a high overlap with R and $[t_b, t_e]$ (e.g., 90% of the documents are covered by R and $[t_b, t_e]$), its pre-trained topic model is often close to the one learnt from the covered documents. Therefore, it is desirable to use the pre-trained topic model of the cell to approximate the topic models learnt from the covered documents, and apply our proposed fast sampling algorithm presented in Section 5.1 to combine it to $\widetilde{\phi}^{(n)}$. In this section, we prove that there exists an error bound parameterized by the overlap using the pre-trained topic model to approximate the one learnt on covered documents.

THEOREM 3. *Let ϕ_k^{ALL} and ϕ_k^{PART} be the language models for topic k learnt on all the documents and the covered documents of a cell, respectively. Let $p \in (0,1)$ be the percentage of documents in the cell that are covered by the input spatial boundary and timespan. Then, the Euclidean distance between two models $\mathbb{E}d(\phi_k^{ALL}, \phi_k^{PART}) < \frac{\sqrt{(1+p^2)}}{p}$.*

PROOF. Let $c_{*,r}^{k,ALL}$ and $c_{*,r}^{k,PART}$ be the counts of assigning topic k to word r in all the documents and the covered documents, respectively. Let $C^{k,ALL} = \sum_r c_{*,r}^{k,ALL}$, $C^{k,PART} = \sum_r c_{*,r}^{k,PART}$. Without loss of generality, we assume the topics of tokens in the covered documents follow similar distribution of those in all the documents in the cell, i.e., $C^{k,PART} \approx pC^{k,ALL}$. Then, the following inequalities hold:

$$\begin{aligned}\mathbb{E}d(\phi_k^{ALL}, \phi_k^{PART})^2 &\approx \mathbb{E}\sum_r\left(\frac{c_{*,r}^{k,ALL} + \beta}{C^{k,ALL} + |V|\beta} - \frac{c_{*,r}^{k,PART} + \beta}{pC^{k,ALL} + |V|\beta}\right)^2 \\ &< \frac{\mathbb{E}(\sum_r c_{*,r}^{k,ALL})^2 + \mathbb{E}(\sum_r c_{*,r}^{k,PART})^2}{(pC^{k,ALL} + |V|\beta)^2}\end{aligned} \tag{9}$$

Because $C^{k,PART} \approx pC^{k,ALL}$, the numerator in Eq. (9) is $(1+p^2)C^{k,PART}$. Thus, we have $\mathbb{E}d(\phi_k^{ALL}, \phi_k^{PART}) < \frac{\sqrt{(1+p^2)}}{p}$. □

According to Theorem 3, we combine the topic models of partially covered (at least τ_p percent of coverage) cells to those of completely covered cells with bounded error. Then for the remaining documents that are covered by the TMRT query, but are not in the cells covered by the query completely or at least τ_p percent, we use online-LDA to sample the topics for them.

5.3 Octree based Pre-Computation

After presenting our techniques for combining pre-computed topic models, we are now ready to present our techniques for the offline pre-computation process. The topic models built in the offline phase will be used to support our online topic mining algorithm.

In the offline pre-computation process, we need to consider two important research problems. First, how to divide the spatio-temporal documents in a large Octree cell into smaller cells to answer TMRT query efficiently and effectively, namely **Octree Cell Division** problem, which affects the accuracy and efficiency of our combining algorithm, and thus the efficiency and effectiveness of answering TMRT queries. Second, how to decide on which Octree cells we pre-train a topic model to balance the efficiency, storage, and accuracy of our approach, namely **Octree Cell Pre-Training** problem. Next, we present our approaches to solving the two problems.

Octree Cell Division: One straightforward way to divide a large Octree cell into smaller ones is to divide it equally in both spatial and temporal dimensions. However, this method is not optimized for our combining algorithm because it does not consider the word overlap between the child cells, which affects the accuracy of our algorithm for combining topic models as given in Theorem 1.

In this paper, we propose to select the division point for each dimension (latitude, longitude and time) by minimizing the word overlap between the 8 child cells divided by the division point. Specifically, suppose D_p be the set of documents in the Octree cell to be divided, $D_p^{(i)}(x)$ be the i-th child cell divided by 3-dimensional point x (latitude, longitude, and time). We optimize the following objective function:

$$\begin{aligned}
&\underset{x}{\text{argmin}} \sum_{i=1}^{8} \sum_{j=1, j\neq i}^{8} |\{\cup_{d\in D_p^{(i)}(x)}\{d.w\}\} \cap \{\cup_{d\in D_p^{(j)}(x)}\{d.w\}\}| \\
&s.t. \quad \frac{|\{d|d^{(h)} > x^{(h)}, d \in D_p\}|}{|D_p|} > \sigma, \\
&\qquad \frac{|\{d|d^{(h)} \leq x^{(h)}, d \in D_p\}|}{|D_p|} > \sigma, \\
&\qquad \forall h \in \{latitude, longitude, time\},
\end{aligned} \tag{10}$$

where $d^{(h)}$ is one of the three dimensions of document d, $x^{(h)}$ is one of the three dimensions of the division point x, and σ is the threshold to control at least how many percentages of documents should be assigned to child cells on both sides of each dimension after division. This treatment is important to balance the size and overlap of the two sub-cubes. An extreme case could happen when we divide the cell into 7 empty cells and a cell that contains all the documents to achieve zero overlap. However, this division is meaningless for accelerating online topic learning. Our treatment can avoid the extreme case. More importantly, our method for Octree cell division can achieve a smaller error bound than the straightforward method of dividing a cell into eight equal-sized cells according to Theorem 1. When the division point of the straightforward method does not satisfy Eq. (10), the overlap between any two sub-cubes is not minimized, and thus the error bound of the straightforward method is larger than that is obtained by the optimal division point in Eq. (10) according to Theorem 1.

We solve this optimization problem as follows. Documents that fall in the cell to be partitioned comprise the set of candidate division points. We traverse these documents and select the point (document) that achieves the minimum objective score defined in Eq. (10). For each candidate point, it takes $O(|W'_p|)$ to compute Eq. (10), where W'_p is the set of word tokens in the documents within the constraints D'_p. Totally we have $|D'_p|$ candidate points, and thus the worst case time complexity is $O(|D'_p||W'_p|)$.

Octree Cell Pre-training: There exist two trade-offs in selecting the cells for topic model pre-computation. First, learning topics for cells in higher levels help improve accuracy of our method by reducing the number of combining operations if the region and timespan in the query cover the high-level cell. Nevertheless, a cell from higher levels is less likely to be completely covered by a user specified region and timespan. TMRT queries with small regions and narrow timespans may not contain the high level cells, and thus we cannot use the pre-trained topic models on high level cells to efficiently answer those TMRT queries. Second, if we build a deep Octree and pre-trained topic models for all cells, there is a better chance we can leverage the pre-trained models by our combining algorithm for a TMRT query. However, learning topic models for all cells in a deep Octree is space and time consuming.

Specifically, to make our online topic combining algorithm efficient and effective, we want to choose a group of Octree cells such that 1) the topic models built on these cells do not exceed a given space threshold τ_s; and 2) it is guaranteed that the error bound of combining topic models in the nearest descendants of these cells does not exceed a given threshold τ_e. This is because even though the TMRT query does not completely cover a high level cell, we can still combine the lower level cells covered by the query without exceeding the error threshold τ_e. To achieve this goal, we propose a greedy division algorithm for building the Octree and select the cells for pre-computation as described in Algorithm 1.

The main idea of the algorithm is to divide the Octree as finer-grained as possible within the space and accuracy threshold. We first insert all the spatio-temporal documents to the root of the Octree (line 1). Each time, we try to divide the largest cell (that contains largest number of spatio-temporal documents) into 8 subregions based on Eq. (10) (lines 5 – 6). To efficiently rank and find the largest cell, we maintain a max heap for cells to be divided with the number of spatio-temporal documents contained in each cell as the key (line 2). We invoke function `UpdateModelPos` (lines 7 – 10) to estimate the number of topic models we need to learn to guarantee the accuracy threshold τ_e. If the number of models exceeds the space threshold τ_s, we withdraw the division of current cell and stop dividing the Octree (lines 8 – 11); Otherwise, we insert the children of the current cell to the max heap to further divide them (lines 12 – 14). We repeat this process until the number of models exceeds the space threshold.

The function `UpdateModelPos` updates the number of models recursively from bottom to top. We aggregate the number of models to be learnt in the descendant cells of the current cell (lines 4 – 5). Suppose that the number of models in the descendants is $subCount$. Then we need $subCount-1$ combining operations to construct a topic model for the current cell. We can measure the error bound of the $subCount-1$ combining operations using a function of the number of combining operations, i.e., $f(subCount-1)$. When the error bound is larger than the accuracy threshold τ_e, we train a topic model for the current cell to ensure that the error does not exceed the accuracy threshold (lines 6 – 8). We introduce an attribute *placeModel* to mark the cells we need to learn a topic model.

After applying Algorithm 1, we traverse the Octree and check the *placeModel* attribute of each cell. If it is true, we learn a topic model for the cell.

Algorithm 1: GreedyDivision

```
Input: A set of spatio-temporal documents $D$, space threshold $\tau_s$,
       error threshold $\tau_e$
Output: Qctree $T$
begin
    Initialize $T$ and Insert $D$ to $T$;
    Initialize max heap $H$, $H$.Insert($T$);
    $modelCount \leftarrow 0$;
    while $modelCount < \tau_s$ do
        $T' \leftarrow H$.DeleteTop();
        $T'$.Subdivide();
        $modelCount \leftarrow$ UpdateModelPos($T$, $\tau_e$);
        if $modelCount > \tau_s$ then
            Withdraw the subdivision for $T'$;
            $modelCount \leftarrow$ UpdateModelPos($T$, $\tau_e$);
            return $T$
        foreach $T_c$ in $T'.Children$ do
            if $T_c.D \cap D \neq \emptyset$ then
                $H$.Insert($T_c$);
    return $T$
Function UpdateModelPos($T$, $\tau_e$)
    $modelCount \leftarrow 0$;
    if $T$ is not leaf then
        $subCount \leftarrow 0$;
        foreach $T_c$ in $T.Children$ do
            $modelCount \leftarrow subCount +$
            UpdateModelPos($T_c$, $\tau_e$);
        if $f(subCount - 1) > \tau_e$ then
            $modelCount \leftarrow modelCount + 1$;
            $T.placeModel \leftarrow$ True;
        else
            $modelCount \leftarrow modelCount + subCount$
            $T.placeModel \leftarrow$ False;
    else if $T.D \cap D \neq \emptyset$ then
        $modelCount \leftarrow 1$;
        $T.placeModel \leftarrow True$;
    return $modelCount$;
```

5.4 Topic Mining within Region and Time Interval

Algorithm 2: OnlineLearning

```
Input: Octree $T$, spatio-temporal query $Q = \{R, [t_b, t_e]\}$, coverage
       threshold $\tau_p \in (0, 1]$
Output: Topic model within $Q$
begin
    $N \leftarrow$ CellSelection($T$, $Q$, $\tau_p$);
    Sort $N$ by descending order of cell size;
    $Model \leftarrow N[0].Model$;
    for $i = 1$ to $N.length$ do
        $Model \leftarrow$ FastSetSampling($Model$, $N[i].Model$);
    $D_{remain} \leftarrow \{d | d \notin N, d \in Q\}$;
    $Model \leftarrow$ OnlineLDA($Model$, $D_{remain}$);
    return $Model$
```

Algorithm 3: CellSelection

```
Input: a Octree $T$, spatio-temporal query $Q = \{R, [t_b, t_e]\}$,
       coverage threshold $\tau_p \in (0, 1]$
Output: cell set $N$
begin
    $N \leftarrow \emptyset$;
    if ($\frac{|T.D \cap Q.D|}{|T.D|} > \tau_p$) and $T$ has pre-trained topic model then
        $N \leftarrow N \cup \{T\}$;
    else if $T.D \cap Q.D \neq \emptyset$ then
        foreach $T_c$ in $T.Children$ do
            $N \leftarrow N \cup$ CellSelection($T_c$);
    return $N$;
```

Given an Octree with pre-trained topic models, we are now ready to present our online topic mining process for a TMRT query with region R and interval $[t_b, t_e]$ in Algorithm 2. We first find the cells that are covered by the given region at least τ_p percent and contain pre-trained topic models (line 1, `CellSelection`). Then, we order the returning cells in the descending order of their size (line 2) because combining smaller cells to larger cells could be more efficient as discussed in Section 5.1. Next, we combine the models of those cells using our fast word set based sampling algorithm (line 5, `FastSetSampling`). Finally, we invoke online-LDA to handle the remaining documents that are not covered by the founded cells.

In Algorithm 2, the function `CellSelection` returns cells that are completely or highly covered by the given TMRT query $Q = \{R, [t_b, t_e]\}$. When we select cells to combine, we aim to reduce the number of combining operations for 1) reducing the total combining time cost, and 2) reducing the error of combining. Therefore, we follow the greedy principle to chooses cells as large as possible (Algorithm 3). The sets $T.D$ and $Q.D$ are the sets of documents covered by the current cell T and query Q, respectively.

In Algorithm 3, we traverse the Octree tree in a breadth-first manner. We add the current cell to the result set if the cell is completely or highly covered by the given query Q (line 2). We continue to search lower levels if the current cell has overlap with Q (lines 4 – 6). We use a predefined coverage threshold τ_p to decide whether a cell is highly covered by Q and the error bound has been proven in Theorem 3.

6. EXPERIMENTAL RESULTS

We introduce the experimental setting in Section 6.1. We report the offline pre-computation cost of our framework in Section 6.2, and the experimental results in Sections 6.3 and 6.4. We show some example topics discovered by our algorithm in Appendix A.2.

6.1 Experimental Setup

Data Sets Our experiments are conducted on three real-life datasets collected from Twitter, Meetup and Wikipedia, respectively. We present our experimental results on Twitter, and Meetup in this section and the results on Wikipedia in Appendix A.1. Dataset Twitter comprises 7 million tweets posted from Aug 2012 to Oct 2012 in United States. Each tweet has text, location (latitude and longitude), and creation time, and each tweet contains 9 words on average. Dataset Meetup comprises 1.1 million events posted from Jan 2006 to Jun 2015. Each event has text description, location and creation time. and each Meetup event description has around 109 words on average. For Twitter, we randomly hold out 20% tweets of each day as test data to evaluate the accuracy of the topic models.

For Meetup, we randomly hold out 20% words of each document as documents in test data to evaluate the accuracy of the topic models. The details of the two datasets are listed in Table 5.

Table 5: Dataset Statistics

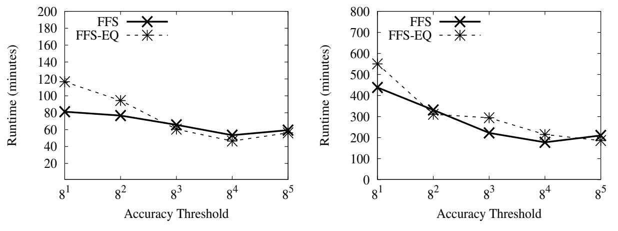
Datasets	Twitter	Meetup
# Documents	7,012,778	1,099,614
# Word Tokens	69,075,864	120,484,706
Vocabulary Size	137,141	88,004
Time Duration	Aug 2012 - Oct 2012	Jan 2006 - Jun 2015

Baseline Methods We compare our method using fast set sampling (FFS) with the following four methods.

LDA: We learn topics for the documents in the given region and timespan using LDA [4] for each TMRT query.

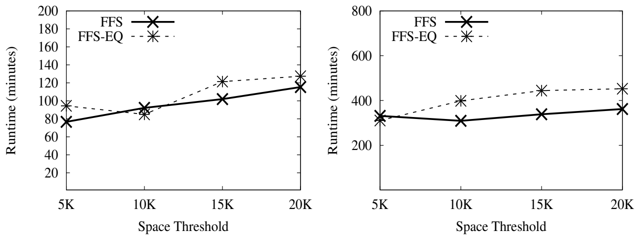
Online-LDA: We leverage the Octree structure in our framework to find the largest cell that is covered by the given region and timespan, and apply online-LDA [2] to sample the remaining documents based on the topic model of the largest cell.

FFS-EQ: We use our fast sampling algorithm to learn topics for TMRT queries. The difference from FFS is that FFS-EQ partitions a cell into 8 qual-sized smaller cells when building the Octree.



(a) Twitter (b) Meetup

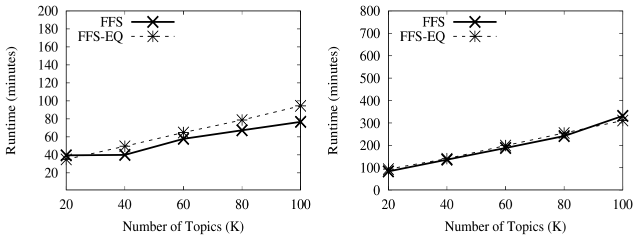
Figure 4: Offline Pre-computation Cost w.r.t. Accuracy Threshold τ_e



(a) Twitter (b) Meetup

Figure 5: Offline Pre-computation Cost w.r.t. Space Threshold τ_s

Evaluation Metrics To evaluate the efficiency and accuracy of the two models, we randomly generate 100 pairs of region and timespan with random size, each containing at least 100 documents in the test set for accuracy evaluation. For efficiency, we report the average runtime of the 100 pairs of region and timespan for all methods. To evaluate accuracy, we use perplexity to measure how likely we can use the model to generate the test set, which is a commonly used measure to evaluate the quality of topic models [25]. Lower perplexity means that the model has better ability of generalization. Perplexity is computed by using the probabilistic model learnt from the training set to estimate the likelihood of generating the test set.



(a) Twitter (b) Meetup

Figure 6: Offline Pre-computation Cost w.r.t. Number of Topics K

Specifically, it is computed as follows:

$$perplexity(W_{test}) = exp\{-\frac{\log p(W_{test}|\Phi_{train})}{|W_{test}|}\}, \qquad (11)$$

where $|W_{test}|$ is the set of word tokens in the test set, and Φ_{train} is the topic model to be evaluated. We apply the "left-to-right" method [25], which is often used for evaluating topic models, to estimate perplexity. We report the sum of perplexity of models learnt from the 100 pairs of region and timespan.

Parameter Settings We have three parameters in our framework, i.e., accuracy threshold τ_e, space threshold τ_s, and number of topics K. Note that the accuracy threshold τ_e is in fact a function of number of cells (n) to combine $f(n)$. For simplicity, we use $\tau_e = f(n)$ to denote the accuracy threshold of combining n cells. The space threshold specifies the maximum number of topic models to pre-train. We analyze the effect of these parameters by varying one parameter while fixing the others as default values. The settings of each parameter is shown in Table 6. The default settings of these parameters are $\tau_e = f(8^2)$ (i.e., learning topic models every 2 levels in the tree), $\tau_s = 5000$ and $K = 100$. We set the constraint for cell division σ in Eq. (10) at 0.3. Online-LDA and FFS-EQ use the same settings as our method (FFS).

All algorithms are implemented in C# on a workstation with Intel(R) Xeon(R) CPU E5-2680 v2 @2.80GHz and 64GB RAM.

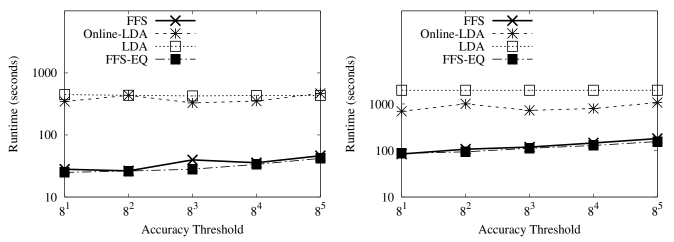
Table 6: Parameter Settings

Parameter (notation)	Settings	Default
accuracy threshold (τ_e)	$f(8^1)$ to $f(8^5)$	$f(8^2)$
space threshold (τ_s)	5K to 20K (topic models)	5K
number of topics (K)	20 to 100	100

6.2 Pre-computation Cost

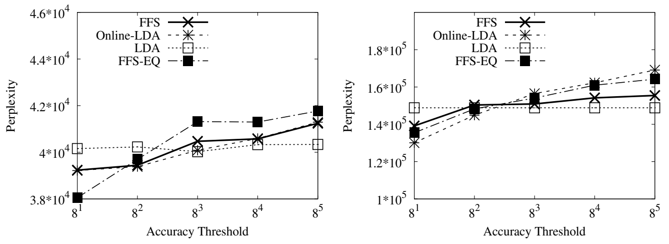
We report the pre-computation cost of FFS-EQ and our method FFS with different parameter settings. From Figure 4, we observe that the pre-computation cost is higher when the accuracy threshold τ_e is set at smaller value (corresponding to smaller error). This is because we need to learn topic models in higher level cells (which contain more documents) in the Octree to achieve better accuracy. Similarly, as the space threshold τ_s or the number of topics increases, the pre-computation costs of the three methods increase as shown in Figure 5 and Figure 6.

Our framework (FFS) with default setting takes around 100 minutes for pre-computation on Twitter data and 300 minutes on Meetup data. The pre-computation time of FFS and FFS-EQ is similar because the computation cost of pre-training topic models overwhelms the computation cost of our cell division method as described in Eq. (10).



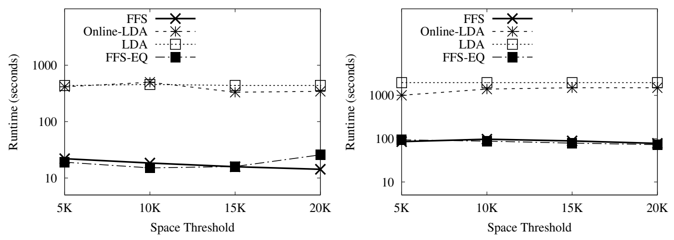
(a) Twitter (b) Meetup

Figure 7: Runtime w.r.t. Accuracy Threshold τ_e



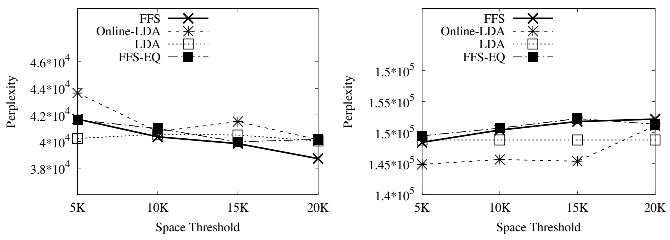
(a) Twitter (b) Meetup

Figure 8: Perplexity w.r.t. Accuracy Threshold τ_e



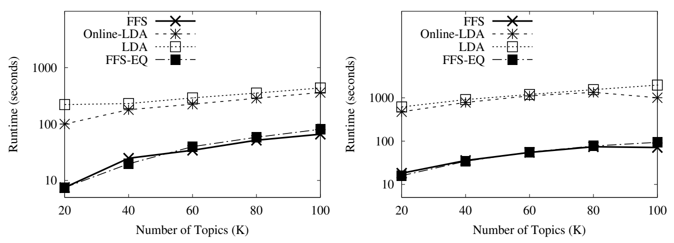
(a) Twitter (b) Meetup

Figure 9: Runtime w.r.t. Space Threshold τ_s



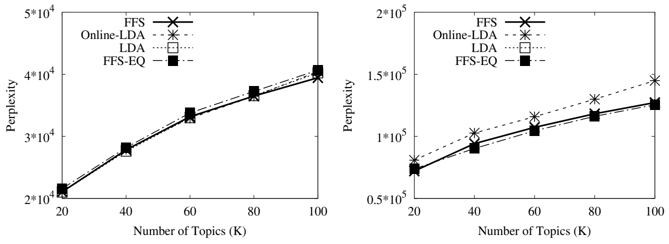
(a) Twitter (b) Meetup

Figure 10: Perplexity w.r.t. Space Threshold τ_s



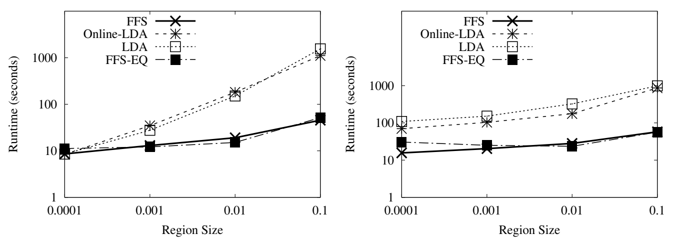
(a) Twitter (b) Meetup

Figure 11: Runtime w.r.t. Number of topics K



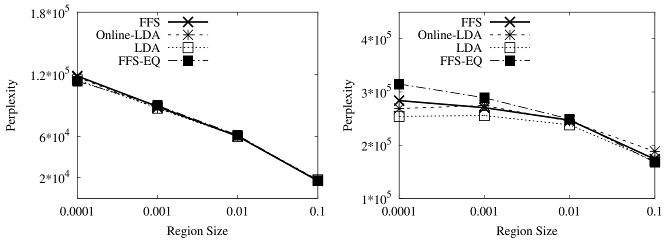
(a) Twitter (b) Meetup

Figure 12: Perplexity w.r.t. Number of Topics K



(a) Twitter (b) Meetup

Figure 13: Runtime w.r.t. Region Size



(a) Twitter (b) Meetup

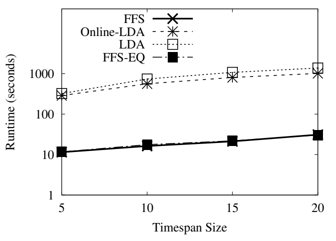
Figure 14: Perplexity w.r.t. Region Size

6.3 Evaluation on Different Parameters

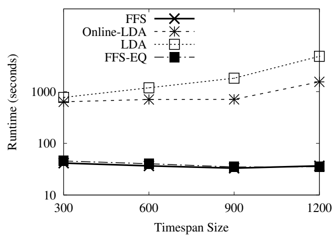
Varying Accuracy Threshold τ_e The accuracy threshold is used to guarantee the error bound of topic combining when we build the Octree as discussed in Section 5.3. We set the threshold as $f(8)$, $f(8^2)$, $f(8^3)$, $f(8^4)$, and $f(8^5)$, which indicates that we pre-train topic models every 1, 2, 3, 4, and 5 levels, respectively. We report the running time of learning topic models for the 100 random pairs of region and timespan in Figure 7(a) and Figure 7(b) as we vary the accuracy threshold.

Our proposed framework FFS is more than an order of magnitude faster than LDA and Online-LDA under different accuracy thresholds because our proposed sampling algorithm can efficiently combine two models without sampling all the word tokens. We also observe that with a smaller accuracy threshold (smaller τ_e), the online topic learning is slightly faster. With a lower accuracy threshold, we learn more pre-trained models for large cells in an Octree. Larger cells are more likely to cover most documents in the specified region and timespan, and thus reduce the time for combining topic models.

Figure 8(a) and Figure 8(b) show the perplexity computed on the test data for all methods. The perplexity of our proposed frame-

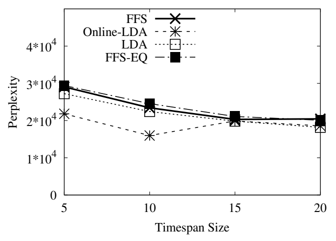


(a) Twitter

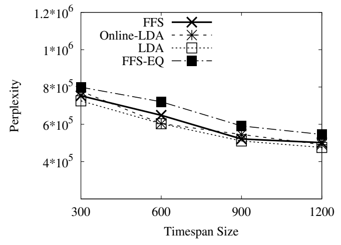


(b) Meetup

Figure 15: Runtime w.r.t. Timespan Size



(a) Twitter



(b) Meetup

Figure 16: Perplexity w.r.t. Timespan Size

work FFS is close to those of LDA and online-LDA because our algorithm combines topic models with a bounded error. Benefiting from the proposed partitioning method of Octree cells, FFS achieves higher accuracy in most cases (lower perplexity) compared with FFS-EQ.

Varying Space Threshold τ_s With a larger space threshold, we have more space for pre-trained topic models, and thus we can construct a deeper Octree. A large space threshold may help accelerate the online learning process because we can learn more pre-trained models in the Octree. As shown in Figure 9(a) and Figure 9(b), FFS and FFS-EQ outperform LDA and online-LDA by more than an order of magnitude. As shown in Figure 10(a) and Figure 10(b), FFS has better accuracy than FFS-EQ in most cases.

Varying Number of topics K The number of topics K affects both Octree based pre-computation and the running time of all methods because they all have complexity linear to K. As shown in Figure 11(a) and Figure 11(b), all methods run slower as K increases. The running time of our framework FFS increases a bit faster than LDA and Online-LDA because the complexity of our framework is quadratic to K. However, in practice K is often set to small values. As shown in Figure 12(a) and Figure 12(b), the perplexity increases when the number of K increases because there are more parameters for topic models (ϕ_k). When the number of parameters increases, the likelihood computed as the joint probability of word tokens decreases.

6.4 Evaluation on Different Regions and Timespans

We also experiment on different user specified regions and timespans. All the parameters are set as default values except the space threshold τ_s is set at 20000. Let the spatial area of the document collection be A. We vary the size of regions with the following values: $0.0001A$, $0.001A$, $0.01A$, $0.1A$ while fixing the size of timespans as 5 days on Twitter data and 300 days on Meetup data. We randomly generate 100 pairs of region and timespan for each of the four settings of region size. The efficiency results are shown in Figure 13(a) and Figure 13(b), and the perplexity results are shown in Figure 14(a) and Figure 14(b). As region size decreases, the running time decreases while the perplexity increases. The running time of our algorithm is close to or even slightly higher than LDA and online-LDA because when the region becomes smaller than the leaf cell of the Octree, our framework is equivalent to LDA. The perplexity of all methods decreases because the number of documents covered by the region increases. Without enough documents (When region size is 0.0001A), each region contains only several hundred or several thousand documents on average for both dataset, the learnt LDA model is less effective on the hold-out test data.

We also vary the timespan by 5, 10, 15, 20 days on Twitter Data and 300, 600, 900, 1200 days on Meetup data while fixing the size of regions to $0.1A$. For each of the timespan, we randomly generate 100 pairs of region and timespan. Figure 15(a) and Figure 15(b) show the efficiency of all methods in different timespans, while Figure 16(a) and Figure 16(b) show the perplexity. With a larger timespan, more documents are included, and thus it takes longer to process. The perplexity of FFS and FFS-EQ is similar to those of LDA and online-LDA as shown in Figure 16(a).

7. CONCLUSIONS AND FUTURE WORK

We study the problem of mining topics from a collection of spatio-temporal documents given a region and a time interval. Based on the LDA model, we propose an algorithm with bounded errors to combine two pre-trained topic models learnt from two document sets, each falling in a region and a time period. We also propose a method to build an Octree based indexing structure to facilitate our online topic learning algorithm. We partition the Octree cells by considering the word overlap of the generated cells to achieve better accuracy for combining two pre-trained topic models. Then we employ our proposed algorithms to combine the pre-trained topic models to obtain the topics for documents in the specified region and time period. Our experimental results suggest that our proposed framework is able to improve the runtime performance by at least an order of magnitude compared with LDA and online-LDA. Meanwhile, the effectiveness of our proposal, measured by perplexity, is close to that of LDA and online-LDA.

This work combines data management principles to improve the efficiency of machine learning/data mining algorithms. This work opens to a number of promising directions for future work. For example, exploratory topic mining from spatio-temporal streams, topic trends detection in a user specified region and timespan, and extended TMRT query with keywords as another query dimension.

Acknowledgment This research was carried out at the Rapid-Rich Object Search (ROSE) Lab at the Nanyang Technological University, Singapore. The ROSE Lab is supported by the National Research Foundation, Prime Minister's Office, Singapore, under its IDM Futures Funding Initiative and administered by the Interactive and Digital Media Programme Office. This work is also supported in part by a Tier-1 grant awarded by Ministry of Education Singapore (RG22/15). We would also like to thank the anonymous reviewers for their constructive comments and suggestions, which help us to improve the paper substantially.

8. REFERENCES

[1] A. Ahmed, M. Aly, J. Gonzalez, S. Narayanamurthy, and A. J. Smola. Scalable inference in latent variable models. In *WSDM*, pages 123–132, 2012.

[2] L. AlSumait, D. Barbará, and C. Domeniconi. On-line LDA: adaptive topic models for mining text streams with

applications to topic detection and tracking. In *ICDM*, pages 3–12, 2008.

[3] A. Angel, N. Koudas, N. Sarkas, and D. Srivastava. What's on the grapevine? In *SIGMOD*, pages 1047–1050, 2009.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[5] B. Ding, B. Zhao, C. X. Lin, J. Han, C. Zhai, A. N. Srivastava, and N. C. Oza. Efficient keyword-based search for top-k cells in text cube. *IEEE Trans. Knowl. Data Eng.*, 23(12):1795–1810, 2011.

[6] W. Feng, C. Zhang, W. Zhang, J. Han, J. Wang, C. Aggarwal, and J. Huang. STREAMCUBE: hierarchical spatio-temporal hashtag clustering for event exploration over the twitter stream. In *ICDE*, pages 1561–1572, 2015.

[7] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000.

[8] T. Hofmann. Learning the similarity of documents: An information-geometric approach to document retrieval and categorization. In *NIPS*, pages 914–920, 1999.

[9] L. Hong, A. Ahmed, S. Gurumurthy, A. J. Smola, and K. Tsioutsiouliklis. Discovering geographical topics in the twitter stream. In *WWW*, pages 769–778, 2012.

[10] L. Hong, G. Convertino, and E. H. Chi. Language matters in twitter: A large scale study. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*, 2011.

[11] G. Li, J. Hu, J. Feng, and K. Tan. Effective location identification from microblogs. In *ICDE*, pages 880–891, 2014.

[12] G. Li, Y. Wang, T. Wang, and J. Feng. Location-aware publish/subscribe. In *KDD*, pages 802–810, 2013.

[13] C. X. Lin, B. Ding, J. Han, F. Zhu, and B. Zhao. Text cube: Computing IR measures for multidimensional text database analysis. In *ICDM*, pages 905–910, 2008.

[14] J. S. Liu. The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem. *Journal of the American Statistical Association*, 89(427):958–966, 1994.

[15] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *SIGMOD*, pages 1155–1158, 2010.

[16] D. Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129–147, 1982.

[17] N. Rehman, S. Mansmann, A. Weiler, and M. Scholl. Building a data warehouse for twitter stream exploration. In *ASONAM*, pages 1341–1348, Aug 2012.

[18] N. Sarkas, A. Angel, N. Koudas, and D. Srivastava. Efficient identification of coupled entities in document collections. In *ICDE*, pages 769–772, 2010.

[19] A. Shraer, M. Gurevich, M. Fontoura, and V. Josifovski. Top-k publish-subscribe for social annotation of news. *PVLDB*, 6(6):385–396, 2013.

[20] A. Simitsis, A. Baid, Y. Sismanis, and B. Reinwald. Multidimensional content exploration. *PVLDB*, 1(1):660–671, 2008.

[21] S. Sizov. Geofolk: Latent spatial semantics in web 2.0 social media. In *WSDM*, pages 281–290, 2010.

[22] A. Smola and S. Narayanamurthy. An architecture for parallel topic models. *PVLDB*, 3(1-2):703–710, Sept. 2010.

[23] J. Strötgen and M. Gertz. Timetrails: A system for exploring spatio-temporal information in documents. *PVLDB*, 3(2):1569–1572, 2010.

[24] J. Strötgen, M. Gertz, and P. Popov. Extraction and exploration of spatio-temporal information in documents. In *GIR*, 2010.

[25] H. M. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno. Evaluation methods for topic models. In *ICML*, pages 1105–1112, 2009.

[26] Y. Wang, H. Bai, M. Stanton, W.-Y. Chen, and E. Y. Chang. Plda: Parallel latent dirichlet allocation for large-scale applications. In *AAIM*, pages 301–314, 2009.

[27] X. Yan, J. Guo, Y. Lan, J. Xu, and X. Cheng. A probabilistic model for bursty topic discovery in microblogs. In *AAAI*, pages 353–359, 2015.

[28] L. Yao, D. Mimno, and A. McCallum. Efficient methods for topic model inference on streaming document collections. In *KDD*, pages 937–946, 2009.

[29] H. Yin, B. Cui, L. Chen, Z. Hu, and Z. Huang. A temporal context-aware model for user behavior modeling in social media systems. In *SIGMOD*, pages 1543–1554, 2014.

[30] Z. Yin, L. Cao, J. Han, C. Zhai, and T. Huang. Geographical topic discovery and comparison. In *WWW*, pages 247–256, 2011.

[31] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Who, where, when and what: Discover spatio-temporal topics for twitter users. In *SIGKDD*, pages 605–613, 2013.

[32] D. Zhang, C. Zhai, and J. Han. Topic cube: Topic modeling for OLAP on multidimensional text databases. In *SDM*, pages 1124–1135, 2009.

[33] D. Zhang, C. Zhai, and J. Han. Mitexcube: Microtextcluster cube for online analysis of text cells. In *CIDU*, pages 204–218, 2011.

[34] D. Zhang, C. Zhai, J. Han, A. N. Srivastava, and N. C. Oza. Topic modeling for OLAP on multidimensional text databases: topic cube and its applications. *Statistical Analysis and Data Mining*, 2(5-6):378–395, 2009.

[35] B. Zhao, C. X. Lin, B. Ding, and J. Han. Texplorer: keyword-based object search and exploration in multidimensional text databases. In *CIKM*, pages 1709–1718, 2011.

[36] K. Zhao, G. Cong, Q. Yuan, and K. Q. Zhu. SAR: A sentiment-aspect-region model for user preference analysis in geo-tagged reviews. In *ICDE*, pages 675–686, 2015.

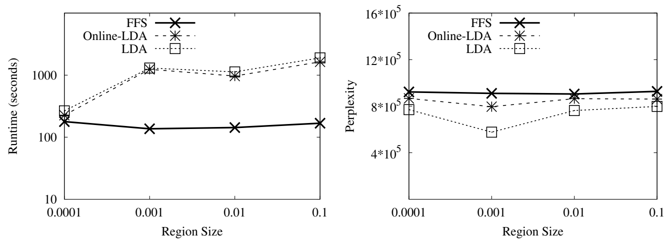
APPENDIX

A. ADDITIONAL EXPERIMENTS

A.1 Evaluation on Wikipedia Data

Wikipedia contains 0.92 million geo-tagged pages, and each page contains 221 words on average. We randomly hold out 20% words of each document as documents in test data to evaluate the perplexity. Different from the two datasets in Section 6.1, Wikipedia pages do not contain temporal information. We replace Octree with Quadtree in our method to answer TMRT queries without timespan. We compare FFS, Online-LDA, and LDA by varying the size of query regions with the following values: $0.0001A$, $0.001A$, $0.01A$, $0.1A$, where A is the spatial area of the Wikipedia document collection. We use the same parameter settings as in Section 6.4. The experimental results are reported in Figure 17.

The three methods have similar efficiencies when the size of the query regions is $0.0001A$. However, as the size of the query regions increases, FFS performs much faster than Online-LDA and LDA, because more documents are covered in larger query region and our proposed method can learn the topic models by combining pre-trained models efficiently. FFS runs around an order of magnitude faster than Online-LDA and LDA, while achieves similar perplexity as the two methods.



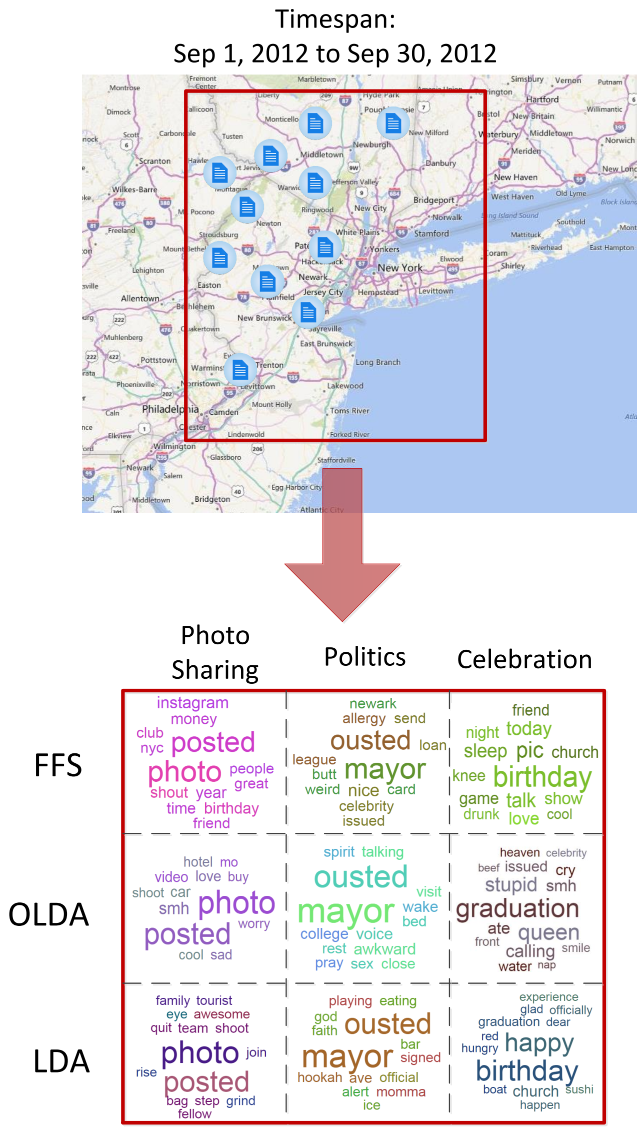
(a) Runtime (b) Perplexity

Figure 17: Runtime and Perplexity w.r.t. Region Size

A.2 Qualitative Evaluation on Extracted Topics

To exemplify the extracted topics of the three methods, we submit a TMRT problem within the region of New York city and timespan from Sep 1, 2012 to Sep 30, 2012, as shown in Figure 18. We apply FFS, LDA and Online-LDA to this TMRT problem using the default parameter setting. The number of words is 637,581, we run 1000 iterations of Gibbs sampling for the three methods. The running time of FFS, LDA and Online-LDA are 104.96 seconds, 1586.14 seconds and 1776.38 seconds, respectively.

We randomly select three topics from LDA and find three topics from the result of online-LDA and our proposed framework (FFS) that are most similar to (measured by KL divergence) the three ones of LDA, respectively. We manually name the extracted three topics as "Photo Sharing", "Politics" and "Celebration", and visualize them using "Word Cloud" in Figure 18.



Timespan:
Sep 1, 2012 to Sep 30, 2012
Photo Sharing
Politics
Celebration
FFS
OLDA
LDA
instagram money club nyc posted photo people great shout year time birthday friend
newark allergy send ousted loan league butt mayor weird nice card celebrity issued
friend night today sleep pic church knee birthday game talk show drunk love cool
hotel mo video love buy shoot car smh photo posted worry cool sad
spirit talking ousted visit mayor wake bed college voice rest awkward pray sex close
heaven celebrity beef issued cry stupid smh graduation ate queen front calling smile water nap
family tourist eye awesome quit team shoot photo join rise posted bag step grind fellow
playing eating god faith ousted bar mayor signed hookah ave official alert momma ice
experience glad officially graduation dear red hungry happy birthday boat church sushi happen

Figure 18: Topics Extracted by Different Methods

In general, the topics extracted by the three methods are similar. Surprisingly, our proposed framework can also discover some representative words for a topic (e.g., "friend" in topic "Photo Sharing" and "work", "education" in topic "Politics") that LDA and online-LDA have not discovered. Consider the quality of extracted topics and runtime efficiency, our proposed framework is more suitable to solve the TMRT problem compared to LDA and online-LDA.