Discovering Diverse and Salient Threads in Document Collections

Jennifer Gillenwater **Alex Kulesza** **Ben Taskar**
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
{jengi,kulesza,taskar}@cis.upenn.edu

Abstract

We propose a novel probabilistic technique for modeling and extracting salient structure from large document collections. As in clustering and topic modeling, our goal is to provide an organizing perspective into otherwise overwhelming amounts of information. We are particularly interested in revealing and exploiting *relationships* between documents. To this end, we focus on extracting diverse sets of *threads*—singly-linked, coherent chains of important documents. To illustrate, we extract research threads from citation graphs and construct timelines from news articles. Our method is highly scalable, running on a corpus of over 30 million words in about four minutes, more than 75 times faster than a dynamic topic model. Finally, the results from our model more closely resemble human news summaries according to several metrics and are also preferred by human judges.

1 Introduction

The increasing availability of large document collections has the potential to revolutionize our ability to understand the world. However, the scale and complexity of such collections frequently make it difficult to quickly grasp the important details and the relationships between them. As a result, automatic interfaces for data navigation, exploration, aggregation, and analysis are becoming increasingly valuable.

In this work we propose a novel approach: *threading* structured document collections. Consider a large graph, with documents as nodes and edges indicating relationships, as in Figure 1. Our goal is to find a diverse set of paths (or threads) through the collection that are individually coherent and together cover the most salient parts. For example, given a collection of academic papers, we might want to identify the most significant lines of research, threading the citation graph to produce chains of important papers. Or, given news articles connected chronologically, we might want to extract threads of articles to form timelines describing the major events from the most significant news stories. Top-tier news organizations like The New York Times and The Guardian regularly publish such timelines, but have so far been limited to creating them by hand. Other possibile applications might include discovering trends on social media sites, or perhaps mining blog entries for important conversations through trackback links. We show how these kinds of threading tasks can be done efficiently, providing a simple, practical tool for representing graph-based data that offers new possibilities compared with existing models.

The Topic Detection and Tracking (TDT) program (Wayne, 2000) has recently led to some research in this direction. Several of TDT's core tasks, like link detection, topic detection, and topic tracking, can be seen as subroutines for the threading problem. Our work, however, addresses these tasks *jointly*, using a global probabilistic model with a tractable inference algorithm. To achieve this, we employ structured determinantal point processes (SDPPs) (Kulesza

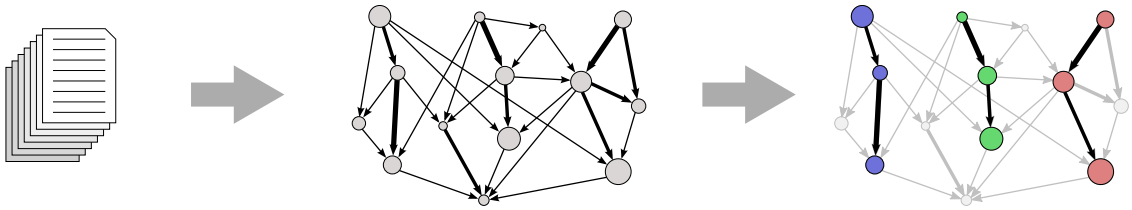


Figure 1: An illustration of document collection threading. We first build a graph from the collection, using measures of importance and relatedness to weight nodes (documents) and build edges (relationships). Then, from this graph, we extract a diverse, salient set of threads to represent the collection. The supplement contains a version of this figure for our real-world news dataset.

and Taskar, 2010), which offer a natural probabilistic model over sets of structures (such as threads) where diversity is desired, and we incorporate k-DPP extensions to control the number of threads (Kulesza and Taskar, 2011).

We apply our model to two real-world datasets, extracting threads of research papers and timelines of news articles. An example of news threads extracted using our model is shown in Figure 2. Quantitative evaluation shows that our model significantly outperforms multiple baselines, including dynamic topic models, in comparisons with human-produced news summaries. It also outperforms baseline methods in a user evaluation of thread coherence, and runs 75 times faster than a dynamic topic model.

The primary contributions of this paper are: (1) proposing a novel framework for finding diverse and salient *sets* of document threads; (2) combining SDPPs and k-DPPs to implement the proposed model; (3) introducing random projections to improve efficiency with only bounded deviation; and (4) demonstrating the model on large-scale, real-world datasets.

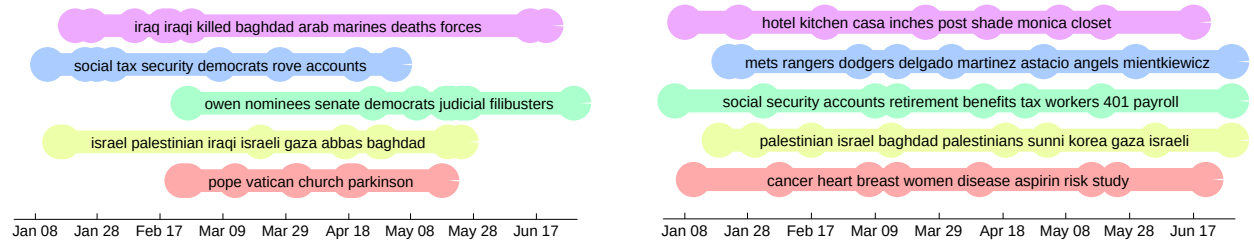
2 Related Work

A variety of papers from the topic tracking literature are broadly related to our work (Mei and Zhai, 2005; Blei and Lafferty, 2006; Leskovec et al., 2009; Ahmed and Xing, 2010). Blei and Lafferty (2006) recently introduced dynamic topic models (DTMs). Assuming a division of documents into time slices, a DTM draws in each slice a set of topics from a Gaussian distribution whose mean is determined by the topics from the previous slice. In this way, a DTM generates *topic* threads. In this work we are interested in the related but not identical task of generating *document* threads. We engineer a baseline for constructing document threads from DTM topic threads (see Section 6.2.2), but the topic-centric nature of DTMs means they are not ideal for this task. Figure 2 illustrates some of the issues.

The work of Ahmed and Xing (2010) generalizes DTMs to iDTMs (infinite DTMs) by allowing topics to span only a subset of time slices, and allowing an arbitrary number of topics. However, iDTMs still require placing documents into discrete epochs, and the issue of generating *topic* rather than *document* threads remains. In Section 6 we compare to DTMs but not iDTMs because an implementation of iDTMs was not readily available.

In the information retrieval community there has also been work on extracting temporal information from document collections. Swan and Jensen (2000) proposed a system for finding temporally clustered named entities in news text and presenting them on a timeline. Allan, Gupta, and Khandelwal (2001) introduced the task of *temporal summarization*, which takes a stream of news articles on a particular topic and tries to extract sentences describing important events as they occur. Yan et al (2011) evaluated methods for choosing sentences from temporally clustered documents that are relevant to a query. Here, we are interested not in extracting topically grouped entities or sentences, but instead in organizing a subset of the articles themselves into timelines, with topic identification as a side effect.

There has also been some prior work focusing more directly on threading. Shahaf and



Feb 24: Parkinson's Disease Increases Risks to Pope
Feb 26: Pope's Health Raises Questions About His Ability to Lead
Mar 13: Pope Returns Home After 18 Days at Hospital
Apr 01: Pope's Condition Worsens as World Prepares for End of Papacy
Apr 02: Pope, Though Gravely Ill, Utters Thanks for Prayers
Apr 18: Europeans Fast Falling Away from Church
Apr 20: In Developing World, Choice [of Pope] Met with Skepticism
May 18: Pope Sends Message with Choice of Name

Jan 11: Study Backs Meat, Colon Tumor Link
Feb 07: Patients—and Many Doctors—Still Don't Know How Often Women Get Heart Disease
Mar 07: Aspirin Therapy Benefits Women, but Not in the Way It Aids Men
Mar 16: Study Shows Radiation Therapy Doesn't Increase Heart Disease Risk for Breast Cancer Patients
Apr 11: Personal Health: Women Struggle for Parity of the Heart
May 16: Black Women More Likely to Die from Breast Cancer
May 24: Studies Bolster Diet, Exercise for Breast Cancer Patients
Jun 21: Another Reason Fish is Good for You

Figure 2: A set of five news threads generated by our method (left) and a dynamic topic model (right) for the first half of 2005. Above, the threads are shown on a timeline with the most salient words superimposed; below, the dates and headlines from the threads appearing at the bottom are listed. Topic models are not designed for threading and often link together topically similar documents that do not constitute a coherent news story, as on the right.

Guestrin (2010) and Chieu and Lee (2004) proposed selecting a *single* thread, whereas we seek a *set* of threads, which is a more general task. Shahaf, Guestrin, and Horvitz (2012) recently proposed *metro maps* as alternative structured representations of related news stories. Metro maps are effectively sets of non-chronological threads that are encouraged to intersect and thus create a "map" of events and topics. However, these approaches assume some prior knowledge about content. Shahaf and Guestrin (2010), for example, assume the thread endpoints are specified, and Chieu and Lee (2004) require a set of query words. These inputs make it possible to quickly pare down the document graph. In constrast, we work with very large graphs and consider all possible threads. Furthermore, while some prior work has relied on heuristics and approximate optimization, we can efficiently sample a joint probabilistic model with approximation guarantees.

In previous work on SDPPs (structured DPPs), which we use here to model threads, Kulesza and Taskar (2010) derived exact polynomial-time algorithms for sampling and other inference. However, their experiments involved feature vectors of only 32 dimensions. For text, natural features like word occurrences typically yield dimensionality in the tens of thousands, making SDPP inference prohibitively expensive. We solve this problem by reducing the feature space using random projections (see Section 5). We prove that even a logarithmic number of projections is sufficient to yield a close approximation to the original SDPP distribution.

3 Framework

Before presenting our probabilistic model, we describe a natural framework for representing document collections. We assume that the collection has been transformed into a directed graph $G = (V, E)$ on n vertices, where each node corresponds to a document and each edge represents a relationship between documents whose semantics depend on the task. We also assume the existence of a weight function w on nodes and edges, which measures the importance or salience of documents and the relative strength of the relationships between them. Formally, we define the weight of a path (or thread) $y = (y^{(1)}, y^{(2)}, \ldots, y^{(T)}), (y^{(t)}, y^{(t+1)}) \in E$ by:

$$w(y) = \sum_{t=1}^{T} w\left(y^{(t)}\right) + \sum_{t=1}^{T-1} w\left(y^{(t)}, y^{(t+1)}\right) \quad . \quad (1)$$

Lastly, we also assume the existence of node features. Specifically, let ϕ represent a feature mapping from nodes to $\mathbb{R}^D$ (for example, tf-idf word vectors). The feature map on a thread is then just a sum over the nodes in the thread:

$$\phi(y) = \sum_{t=1}^{T} \phi\left(y^{(t)}\right) \ . \tag{2}$$

(If it is convenient to have features on edges as well as on nodes, it is possible to accommodate them without affecting asymptotic performance.) Given this framework, our goal is to develop a probabilistic model over sets of k threads of length T, favoring sets whose threads have large weight but are also distinct from one another with respect to ϕ. In other words, a high-probability set under the model should include threads that are both salient and diverse.

This is a daunting problem, given that the number of possible sets of threads is $O(n^{kT})$. For the datasets we use later, the actual number is around 2^{1000}. However, we will show how to construct the desired model in a way that allows efficient inference, even for large datasets, using determinantal point processes (DPPs). We begin with some background.

4 Determinantal point processes

A DPP is a type of distribution over subsets. Formally, a DPP $\mathcal{P}$ on a set of items $\mathcal{Y} = \{y_1, \ldots, y_N\}$ is a probability measure on $2^{\mathcal{Y}}$, the set of all subsets of $\mathcal{Y}$. (In our setting, $\mathcal{Y}$ will be the set of all possible threads.) For every $Y \subseteq \mathcal{Y}$ we have:

$$\mathcal{P}(Y) = \frac{\det(L_Y)}{\sum_{Y \subseteq \mathcal{Y}} \det(L_Y)} = \frac{\det(L_Y)}{\det(L+I)} \ , \tag{3}$$

where L is a positive semidefinite matrix and I is the $N \times N$ identity matrix. $L_Y \equiv [L_{ij}]_{y_i y_j \in Y}$ denotes the restriction of L to the entries indexed by elements of Y, and $\det(L_\emptyset) = 1$. We can define the entries of L as follows:

$$L_{ij} = q(y_i)\phi(y_i)^\top \phi(y_j) q(y_j) \ , \tag{4}$$

where we can think of $q(y_i) \in \mathbb{R}^+$ as the "quality" of an item y_i, and $\phi(y_i) \in \mathbb{R}^D$, $\|\phi(y_i)\|_2 = 1$ as a normalized D-dimensional feature vector such that $\phi(y_i)^\top \phi(y_j) \in [-1, 1]$ is a measure of similarity between items y_i and y_j. This simple definition gives rise to a distribution that places most of its weight on sets that are both high quality and diverse. To understand why this is the case, note that determinants are closely related to volumes; in particular, $\det(L_Y)$ is proportional to the volume spanned by the vectors $q(y_i)\phi(y_i)$ for $y_i \in Y$. Thus, sets with high-quality, diverse items have the highest probability; see Figure 3 for an illustration.

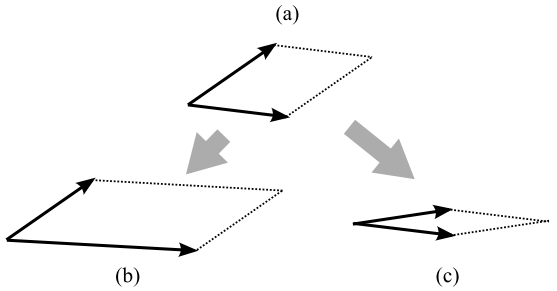


Figure 3: (a) The DPP probability of a set Y depends on the volume spanned by vectors $q(y_i)\phi(y_i)$ for $i \in Y$. (b) As quality (length) increases, so does volume. (c) As similarity increases, volume decreases.

4.1 Structured DPPs

Kulesza and Taskar (2010) introduced *structured* DPPs (SDPPs) to efficiently handle $\mathcal{Y}$ containing exponentially many structures. In our setting, $\mathcal{Y}$ contains all threads of length T, so each $y_i \in \mathcal{Y}$ is a sequence $(y_i^{(1)}, \ldots, y_i^{(T)})$, where $y_i^{(t)}$ is the document included in the thread at position t. When G is a complete graph, there are n^T possible sequences, so $|\mathcal{Y}| = N = n^T$.

In order to allow for efficient normalization and sampling, SDPPs assume a factorization of the quality score $q(y_i)$ and similarity score $\phi(y_i)^\top \phi(y_j)$ into parts, decomposing quality multiplicatively and similarity additively:

$$q(y_i) = \prod_{t=1}^{T} q\left(y_i^{(t)}\right) \quad \phi(y_i) = \sum_{t=1}^{T} \phi\left(y_i^{(t)}\right) \tag{5}$$

For threading, the definition of ϕ is just as given in Equation (2). However, in order to convert the weight function defined in Equation (1) to the

appropriate multiplicative form, we use a simple log-linear model, setting $q(y_i) = \exp(\lambda w(y_i))$, where λ is a hyperparameter that effectively governs the balance between quality and diversity by adjusting the dynamic range of the quality function.

An efficient algorithm for sampling structures (in this case, sets of threads) from an SDPP is derived in Kulesza and Taskar (2010). While the details are beyond the scope of this paper, we note that the sampling algorithm requires $O(Tn^2D^2)$ time. If the node degrees are bounded by r then the time is reduced to $O(TrnD^2)$. This is not quite efficient enough when the number of features, D, is large, as it often is for textual tasks, but we will show in Section 5 how to overcome this last hurdle.

Note that, in our later experiments, we fix T to moderate values ($T = 5, 8$) for ease of analysis and display. However, it is possible (and efficient, due to the linear scaling) to allow longer threads, as well as threads of variable length. The latter effect can be achieved by adding a single "dummy" node to the document graph, with incoming edges from all other documents and a single outgoing self-loop edge. Shorter threads will simply transition to this dummy node when they are complete.

4.2 k-DPPs

SDPPs allow us to efficiently model all sets of threads; however, for practical reasons we would prefer to focus only on sets of exactly k threads. To do so we exploit recently developed methods for working with DPPs of fixed size (Kulesza and Taskar, 2011). A k-DPP $\mathcal{P}^k$ is a DPP conditioned on the event that the subset $Y \in \mathcal{Y}$ has cardinality k; formally, whenever $|Y| = k$:

$$\mathcal{P}^k(Y) = \frac{\det(L_Y)}{\sum_{|Y'|=k} \det(L_{Y'})} \; . \tag{6}$$

In this work we combine k-DPPs with SDPPs, referring to the result as a k-SDPP. We note that using k-SDPPs instead of SDPPs does not affect efficiency of sampling; it merely affords a mechanism for controlling the number of threads.

5 Random projections

As described above, the time complexity for sampling sets from SDPPs is $O(TrnD^2)$. Although this is polynomial, for practical problems nD^2 is prohibitively large. While previous work has dealt only with small datasets, in our experiments we typically have $n, D > 30{,}000$; storing a single message for the message-passing routine involved in SDPP sampling would require over 200 terabytes of memory. To make the model practical, therefore, we turn to techniques for dimensionality reduction.

Standard PCA requires $O(D^3)$ time and would be much too slow. But a classic result of Johnson and Lindenstrauss (1984) shows that high-dimensional points can be *randomly* projected onto a logarithmic number of dimensions while approximately preserving the distances between them. More recently, Magen and Zouzias (2008) extended this idea to the preservation of volumes spanned by sets of points. Here, we use a relationship between determinants and volumes to adapt the latter result. We will prove the following bound on the variational distance between the original k-SDPP and a randomly projected version.

Theorem 1. *Fix $\epsilon, \delta < 1/2$, and set*

$$d = \max\left\{\frac{2k}{\epsilon}, \frac{24}{\epsilon^2}\left(\frac{\log(3/\delta)}{\log N} + 1\right)\log 2N + k - 1\right\}. \tag{7}$$

Let $\mathcal{P}^k$ be the k-SDPP distribution in Equation (6), let G be a $d \times D$ random matrix whose entries are independently sampled from $\mathcal{N}(0, 1/d)$, and let $\tilde{\mathcal{P}}^k(Y)$ be the k-SDPP distribution after projecting ϕ by G—that is, replacing ϕ with $G\phi$. Then with probability at least $1 - \delta$,

$$\|\mathcal{P}^k - \tilde{\mathcal{P}}^k\|_1 = \sum_{|Y|=k} |\mathcal{P}^k(Y) - \tilde{\mathcal{P}}^k(Y)| \leq e^{6k\epsilon} - 1 \; . \tag{8}$$

Note that $e^{6k\epsilon} - 1 \approx 6k\epsilon$ when $k\epsilon$ is small, and $d = O(\max\{k/\epsilon, (\log(1/\delta) + T\log n)/\epsilon^2\})$.

Practically, Theorem 1 says that if we project ϕ down to dimension d logarithmic in the number of documents and linear in thread length, the L_1 variational distance between the true model and the projected model is bounded.

To prove Theorem 1, we will first state a variant of Magen and Zouzias' result, which bounds the ratio of volumes before and after projection from D down to d dimensions.

Lemma 1. *Let X be a $D \times N$ matrix. Fix $k < N$ and $\epsilon, \delta < 1/2$, and set d and G as in Theorem 1. Then with probability at least $1 - \delta$ we have, for all $D \times k$ matrices Y formed by a subset of k columns from X:*

$$(1-\epsilon)^k \leq \frac{\mathrm{Vol}(GY)}{\mathrm{Vol}(Y)} \leq (1+\epsilon)^k \ ,$$

where $\mathrm{Vol}(Y)$ is the k-dimensional volume spanned by the columns of Y and the origin.

We can make use of the following fact to convert this bound on volumes to a bound on determinants:

$$\mathrm{Vol}(Y) = \frac{1}{k!}\sqrt{\det(Y^\top Y)} \ . \tag{9}$$

In order to handle the k-SDPP normalization constant

$$\sum_{|Y|=k} \left(\prod_{y_i \in Y} q^2(y_i) \right) \det(\phi(Y)^\top \phi(Y)) \ , \tag{10}$$

we also must adapt Lemma 1 to *sums* of determinants. The following lemma gives the details.

Lemma 2. *Under the same conditions as Lemma 1, with probability at least $1 - \delta$,*

$$(1+2\epsilon)^{-2k} \leq \frac{\sum_{|Y|=k} \det((GY)^\top (GY))}{\sum_{|Y|=k} \det(Y^\top Y)} \leq (1+\epsilon)^{2k} \ .$$

Proof.

$$\begin{aligned}
\sum_{|Y|=k} \det((GY)^\top (GY)) &= \sum_{|Y|=k} (k!\mathrm{Vol}(GY))^2 \\
&\geq \sum_{|Y|=k} \left(k!\mathrm{Vol}(Y)(1-\epsilon)^k\right)^2 \\
&\geq (1+2\epsilon)^{-2k} \sum_{|Y|=k} \det(Y^\top Y) \ ,
\end{aligned}$$

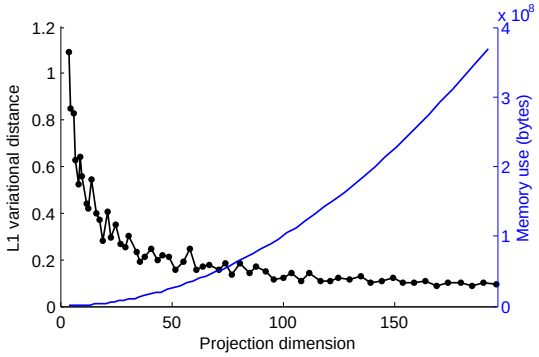


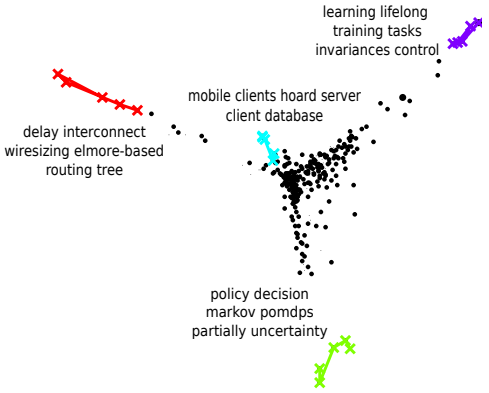
Figure 4: The effect of random projections. In black, on the left, we estimate the L_1 variational distance between the true and projected models. In blue, on the right, we plot the memory required for sampling. Running time is proportional to memory use.

where the first inequality holds with probability at least $1-\delta$ by Lemma 1, and the second follows from the fact that $(1-\epsilon)(1+2\epsilon) \geq 1$ (since $\epsilon < 1/2$), thus $(1-\epsilon)^{2k} \geq (1+2\epsilon)^{-2k}$. A symmetric argument gives the upper bound. □

Proof (of Theorem 1). Let B be the matrix whose columns are given by $B_i = q(y_i)\phi(y_i)$. We have

$$\begin{aligned}
\|\mathcal{P}^k - \tilde{\mathcal{P}}^k\|_1 &= \sum_{|Y|=k} |\mathcal{P}^k(Y) - \tilde{\mathcal{P}}^k(Y)| \\
&= \sum_{|Y|=k} \mathcal{P}^k(Y) \left| 1 - \frac{\tilde{\mathcal{P}}^k(Y)}{\mathcal{P}^k(Y)} \right| \\
&= \sum_{|Y|=k} \mathcal{P}^k(Y) \left| 1 - \frac{\det([GB_Y^\top][GB_Y])}{\det(B_Y^\top B_Y)} \right. \\
&\qquad \left. \cdot \frac{\sum_{|Y'|=k} \det(B_{Y'}^\top B_{Y'})}{\sum_{|Y'|=k} \det([GB_{Y'}^\top][GB_{Y'}])} \right| \\
&\leq \left| 1 - (1+\epsilon)^{2k}(1+2\epsilon)^{2k} \right| \sum_{|Y|=k} \mathcal{P}^k(Y) \\
&\leq e^{6k\epsilon} - 1 \ ,
\end{aligned}$$

where the first inequality follows from Lemma 1 and Lemma 2, which hold simultaneously with probability at least $1-\delta$, and the second follows from $(1+a)^b \leq e^{ab}$ for $a, b \geq 0$. □



learning lifelong training tasks invariances control

- Locally Weighted Learning for Control
- Discovering Structure in Multiple Learning Tasks: The TC Algorithm
- Learning One More Thing
- Explanation Based Learning for Mobile Robot Perception
- Learning Analytically and Inductively

mobile clients hoard server client database

- A Database Architecture for Handling Mobile Clients
- An Architecture for Mobile Databases
- Database Server Organization for Handling Mobile Clients
- Mobile Wireless Computing: Solutions and Challenges in Data Management
- Energy Efficient Query Optimization

Figure 5: Example threads sampled from a 4-SDPP with thread length $T = 5$ on the Cora dataset. We project from word-space to two dimensions by running PCA on the centroids of the threads. The nodes not on the thread paths form a representative subset of the other documents from Cora. Displayed beside each thread are a few of its maximum-tfidf words. Paper titles from two of the threads are shown to the right.

6 Experiments

We begin by showing the performance of random projections on a small, synthetic threading task where the exact model is tractable, with $n = 600$ and $D = 150$. Figure 4 shows the L_1 variational distance (estimated by sampling) as well as the actual memory required for a variety of projection dimensions d. Note that, as predicted by Theorem 1, fidelity to the true model increases rapidly with d.

6.1 Cora citation graph

To qualitatively illustrate our model, we apply it to Cora (McCallum et al., 2000). Cora is a large collection of academic papers on computer science topics, plus citations between them. We construct a directed graph with papers as nodes and citations as edges; after removing papers with missing metadata or zero outgoing citations, our graph contains $n = 28{,}155$ papers.

To obtain useful threads, we set edge weights to reflect the degree of textual similarity between the citing and cited papers, and set node weights to reflect paper "importance". Edge weights are given by normalized cosine similarity (NCS), which for two documents i and j is the dot product of their normalized tfidf vectors:

$$\frac{\sum_{w \in W} \text{tfidf}_i(w)\text{tfidf}_j(w)}{\sqrt{\sum_{w \in W} \text{tfidf}_i(w)^2}\sqrt{\sum_{w \in W} \text{tfidf}_j(w)^2}},$$

where W is a subset of the words found in the documents. We select W by filtering according to document frequency; that is, we remove words that are too common or too rare. After filtering, there are 50,912 unique words. The node weights are given by LexRank scores (Erkan and Radev, 2004), which are similar to node degrees.

Finally, we build a similarity feature map ϕ to encourage diversity. We represent each document by the 1000 documents to which it is most similar according to NCS; this results in binary ϕ of dimension $m = n$ with exactly 1000 non-zeros. The dot product between the similarity features of two documents is thus proportional to the fraction of top-1000 similar documents they have in common. As described in Section 5, we then randomly project this large feature set from $D \approx 28{,}000$ to $d = 50$ dimensions.

We illustrate the behavior of the resulting model in Figure 5. The discovered threads occupy distinct regions of word-space, standing apart visually, and contain diverse salient terms.

6.2 News articles

For quantitative evaluation, we use newswire data. Our dataset comprises over 200,000 articles from the New York Times, collected from 2005-2007 as part of the English Gigaword corpus (Graff and Cieri, 2009). We split the articles into six-month time periods, with an average of

$n = 34{,}504$ articles per period. After filtering, there are a total of 36,356 unique words.

For each time period, we generate a graph with articles as nodes. We use NCS for edge weights, and throw away edges with weight < 0.1. We also require that edges go forward in time; this enforces the chronological ordering of our threads. The supplement contains illustrations of one of the resulting graphs. We use LexRank for node weights and the top-1000 similar documents as similarity features ϕ, projecting to $d = 50$, as before (Section 6.1). We also add a constant feature ρ to ϕ, which controls the overall degree of repulsion; large values of ρ make all documents more similar. This makes the k-SDPP distribution more peaked around diverse sets. For all of the following results, we use $T = 8$ and $k = 10$ so that the resulting timelines are of a manageable size for analysis. However, we tried several values of k and T in our experiments, and did not see significant differences in relative performance. We report all metrics averaged over 100 random samples from the model for each six-month period.

6.2.1 Graph visualizations

The (very large) news graph for the first half of 2005 can be viewed interactively at http://zoom.it/jOKV. In this graph each node (dark circle) represents a news article, and is annotated with its headline. Node size corresponds to weight (LexRank score). Nodes are laid out chronologically, left-to-right, from January to June of 2005. The five colored paths indicate a set of threads sampled from the k-SDPP. Headlines of the articles in each thread are colored to match the thread. Edges are included as described in the paper, but due to the scale of this dataset, only 1% of the edges are shown. Edge thickness corresponds to weight (NCS).

We provide a view of a small subgraph for illustration purposes in Figure 6, which shows the incoming and outgoing edges for a single node. A zoomable version of this subgraph is available at http://zoom.it/GUCR.

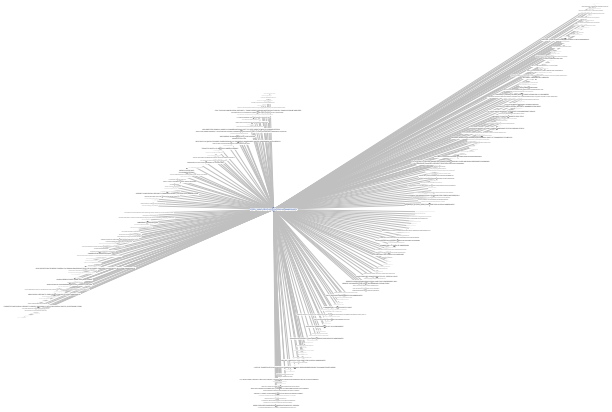


Figure 6: Snapshot of a single article node and all of its neghboring article nodes. See http://zoom.it/GUCR for the zoomable image.

6.2.2 Baselines

k-means baseline: A simple baseline is to split each six-month period of articles into T equal time slices, then apply k-means clustering to each slice, using NCS to measure distance. We then select the most central article from each cluster, and finally match the k articles from time slice i one-to-one with those from slice $i + 1$ by computing the pairing that maximizes the average NCS of the pairs, i.e., the coherence of the threads. The result is a set of k threads of length T, where no two threads contain the same article. In its use of clustering, this baseline is somewhat similar to the "event threading" baseline of Shahaf and Guestrin (2010).

DTM baseline: A more sophisticated baseline is the dynamic topic model (Blei and Lafferty, 2006), which explicitly attempts to find topics that are smooth through time. We use code provided by the authors to fit DTMs with the number of topics set to k and with the data split into T equal slices, as before. We then choose, for each topic at each time step, the document with the highest per-word probability of being generated by that topic. Documents from the same topic form a single thread.

	CosSim	ROUGE-1		ROUGE-2		ROUGE-SU4	
		F	Prec / Rec	F	Prec / Rec	F	Prec / Rec
k-means	29.9	16.5	17.3 / 15.8	0.695	0.725 / 0.669	3.76	3.94 / 3.60
DTM	27.0	14.7	15.5 / 14.0	0.750	0.813 / 0.698	3.44	3.63 / 3.28
k-SDPP	**33.2**	**17.2**	**17.7/16.7**	**0.892**	**0.917/0.870**	**3.98**	**4.11/3.87**

Table 1: Similarity of automatically generated timelines to human summaries. Bold entries are significantly higher than others in the column at 99% confidence, computed using bootstrapping (Hesterberg et al., 2003).

6.2.3 Comparison to human summaries

We compare the threads generated by our baselines and sampled from the k-SDPP to a set of human-generated news summaries. The human summaries are not threaded; they are flat, roughly daily news summaries published by Agence France-Presse and found in the Gigaword corpus, distinguished by their "multi" type tag. A sample summary is included in the supplement. These summaries tend to focus on world news, which is only a subset of the contents of our dataset. However, they allow us to provide an extrinsic evaluation of our method without gold standard timelines. We compute four statistics:

- **Cosine similarity**: NCS (in percent) between the concatenated threads and concatenated human summaries. The hyperparameters for all methods—such as the constant feature magnitude ρ for k-SDPPs and the parameter governing topic proportions for DTMs—were tuned to optimize cosine similarity on a development set from January-June 2005.
- **ROUGE-1**, **2**, and **SU4**: Standard ROUGE scores for summarization evaluation (Lin, 2004).

Table 1 shows the results of these comparisons, averaged across all six half-year intervals. Under each measure, the k-SDPP threads more closely resemble human summaries.

6.2.4 Mechanical Turk evaluation

An important distinction between the baselines and the k-SDPP is that the former are *topic*-oriented, choosing articles that relate to broad subject areas, while our approach is *story*-oriented, chaining together articles with direct individual relationships. An example of this distinction can be seen in Figure 2.

	Rating	Interlopers
k-means	2.73	0.71
DTM	3.19	1.10
k-SDPP	**3.31**	1.15

Table 2: Rating: average coherence score from 1 (worst) to 5 (best). Interlopers: average number of interloper articles identified (out of 2). Bold entries are significantly higher with 95% confidence.

To obtain a large-scale evaluation of thread coherence, we turn to Mechanical Turk. We asked Turkers to read the headlines and first few sentences of each article in a timeline and then rate the overall narrative coherence of the timeline on a scale of 1 ("the articles are totally unrelated") to 5 ("the articles tell a single clear story"). Five separate Turkers rated each timeline; the average ratings are shown in Table 2. Note that k-means does particularly poorly in terms of coherence since it has no way to ensure that clusters are similar between time slices.

We also had Turkers evaluate threads implicitly by performing a simple task. We showed them timelines into which two additional "interloper" articles selected at random had been inserted, and asked them to remove the two articles that they thought should be removed to "improve the flow of the timeline". A screenshot of the task is provided in the supplement. Intuitively, the interlopers should be selected more often when the original timeline is coherent. The average number of interloper articles correctly identified is shown in Table 2.

	Runtime
k-means	625.63
DTM	19,433.80
k-SDPP	**252.38**

Table 3: Time (in seconds) required to produce a complete set of threads. The test machine has eight Intel Xeon E5450 cores and 32GB of memory.

6.2.5 Runtimes

Finally, we report in Table 3 the time required to produce a complete set of threads for each method. This time includes clustering for k-means, model fitting for DTM and random projections, computation of the covariance matrix, and sampling for k-SDPP. We view the graph as an input (much like tfidf vectors for the baselines), and so do not include its computation in the runtime for the k-SDPP. Constructing the graph only requires an additional 160 seconds though.

6.3 Analysis

Below we briefly summarize the main differences between the k-SDPP and the baselines, and discuss their significance.

- Neither baseline *directly* models the document threads themselves. In contrast, the k-SDPP defines a probability distribution over all possible sets of document threads. This makes the k-SDPP a better choice for applications where, for instance, the coherence of individual threads is important.

- While the baselines seek threads that cover or explain as much of the dataset as possible, k-SDPPs are better suited for tasks where a balance between quality and diversity is key, since its hyperparameters correspond to weights on these quantities. With news timelines, for example, we want not just topical diversity but also a focus on the most important stories.

- Both baselines require input to be split into time slices, whereas the k-SDPP does not; this flexibility allows the k-SDPP to put multiple articles from a single time slice in a thread, or to build threads that span only part of the input period.

- While clustering and topic models rely on EM to approximately optimize their objectives, the k-SDPP comes with an exact, polynomial-time sampling algorithm.

Revisiting Figure 2, we can see all of these advantages in action. The k-SDPP produces more consistent threads due to its use of graph information, while the DTM threads, though topic-focused, are less coherent as a story. Furthermore, DTM threads span the entire time period, while our method selects threads covering only relevant spans. The quantitative results in this section underscore the empirical value of these characteristics.

7 Conclusion

We introduced the novel problem of finding diverse and salient threads in graphs of large document collections. We developed a probabilistic approach, combining SDPPs and k-SDPPs, and showed how random projections make inference efficient and yield an approximate model with bounded variational distance to the original. We then demonstrated that the method produces qualitatively reasonable results, and, relative to several baslines, reproduces human news summaries more faithfully, builds more coherent story threads, and is significantly faster. It would be interesting to extend our model to structures beyond linear chains to trees and other structures.

8 Acknowledgements

This material is based upon work supported under a National Science Foundation Graduate Research Fellowship and NSF award 0803256.

References

[Ahmed and Xing2010] A. Ahmed and E. Xing. 2010. Timeline: A Dynamic Hierarchical Dirichlet Process Model for Recovering Birth/Death and Evolution of Topics in Text Stream. In *Proc. UAI.*

[Allan et al.2001] J. Allan, R. Gupta, and V. Khandelwal. 2001. Temporal Summaries of New Topics. In *Proc. SIGIR.*

[Blei and Lafferty2006] D. Blei and J. Lafferty. 2006. Dynamic Topic Models. In *Proc. ICML*.

[Chieu and Lee2004] H. Chieu and Y. Lee. 2004. Query Based Event Extraction along a Timeline. In *Proc. SIGIR*.

[Erkan and Radev2004] G. Erkan and D.R. Radev. 2004. LexRank: Graph-Based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research*, 22(1):457–479.

[Graff and Cieri2009] D. Graff and C. Cieri. 2009. English Gigaword.

[Hesterberg et al.2003] T. Hesterberg, S. Monaghan, D. Moore, A. Clipson, and R. Epstein. 2003. *Bootstrap Methods and Permutation Tests*.

[Johnson and Lindenstrauss1984] W. B. Johnson and J. Lindenstrauss. 1984. Extensions of Lipschitz Mappings into a Hilbert Space. *Contemporary Mathematics*, 26:189–206.

[Kulesza and Taskar2010] A. Kulesza and B. Taskar. 2010. Structured Determinantal Point Processes. In *Proc. NIPS*.

[Kulesza and Taskar2011] A. Kulesza and B. Taskar. 2011. k-DPPs: Fixed-Size Determinantal Point Processes. In *Proc. ICML*.

[Leskovec et al.2009] J. Leskovec, L. Backstrom, and J. Kleinberg. 2009. Meme-tracking and the Dynamics of the News Cycle. In *Proc. KDD*.

[Lin2004] C.Y. Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. WAS*.

[Magen and Zouzias2008] A. Magen and A. Zouzias. 2008. Near Optimal Dimensionality Reductions that Preserve Volumes. *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 523–534.

[McCallum et al.2000] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. 2000. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval Journal*, 3:127–163.

[Mei and Zhai2005] W. Mei and C. Zhai. 2005. Discovering Evolutionary Theme Patterns From Text: An Exploration of Temporal Text Mining. In *Proc. KDD*.

[Shahaf and Guestrin2010] D. Shahaf and C. Guestrin. 2010. Connecting the Dots Between News Articles. In *Proc. KDD*.

[Shahaf et al.2012] D. Shahaf, C. Guestrin, and E. Horvitz. 2012. Trains of Thought: Generating Information Maps. In *Proc. WWW*.

[Swan and Jensen2000] R. Swan and D. Jensen. 2000. TimeMines: Constructing Timelines with Statistical Models of Word Usage. In *Proc. KDD*.

[Wayne2000] C. Wayne. 2000. Multilingual Topic Detection and Tracking: Successful Research Enabled by Corpora and Evaluation. In *Proc. LREC*.

[Yan et al.2011] R. Yan, X. Wan, J. Otterbacher, L. Kong, X. Li, and Y. Zhang. 2011. Evolutionary Timeline Summarization: A Balanced Optimization Framework via Iterative Substitution. In *Proc. SIGIR*.