Mining Blog Stories Using Community-Based and Temporal Clustering

Arun Qamra
Dept. of Computer Science
UC Santa Barbara
arun@cs.ucsb.edu

Belle Tseng
NEC Labs America
Cupertino
belle@sv.nec-labs.com

Edward Y. Chang
Elec. and Comp. Engg.
UC Santa Barbara
echang@ece.ucsb.edu

ABSTRACT

In recent years, weblogs, or blogs for short, have become an important form of online content. The personal nature of blogs, online interactions between bloggers, and the temporal nature of blog entries, differentiate blogs from other kinds of Web content. Bloggers interact with each other by linking to each other's posts, thus forming online communities. Within these communities, bloggers engage in discussions of certain issues, through entries in their blogs. Since these discussions are often initiated in response to online or offline events, a discussion typically lasts for a limited time duration. We wish to extract such temporal discussions, or *stories*, occurring within blogger communities, based on some query keywords. We propose a *Content-Community-Time* model that can leverage the content of entries, their timestamps, and the community structure of the blogs, to automatically discover stories. Doing so also allows us to discover *hot* stories. We demonstrate the effectiveness of our model through several case studies using real-world data collected from the blogosphere.

Categories and Subject Descriptors: H.2.8 [Database Management]: Database Applications - Data Mining

General Terms: Algorithms, Experimentation

Keywords: Weblogs, Online-Communities, Time-sensitive Clustering

1. INTRODUCTION

In the past few years, weblogs, or blogs as they are more commonly known, have emerged as an important type of Web page. Though blogs started out as online diaries, easy-to-use blogging tools (Blogger [4], LiveJournal [18] etc.) have led to an explosion in their number, and blogs have evolved into an important medium for online content. Several million blogs are currently in existence. Typically, a blog is a Web page comprised of a sequence of dated entries.

Some characteristics make blogs different from other Web pages. First, each blog is usually maintained and updated by one person (called the *blogger*). Hence, blogs are personal, representing the interests, opinions and interactions of one blogger. Second, blogs are updated regularly with new entries; hence they contain rapidly evolving content, as compared to regular websites that do not change much over time. Third, each entry in a blog has an associated time stamp, unlike regular webpages. Also, bloggers interact with each other by linking to each other's entries in their own entries, thus forming online communities. Within these communities, bloggers engage in discussions over certain issues, through entries in their blogs. These discussions, or *stories* in the *blogosphere* exist for a while and then die out.

Because of the increasing number of blogs and their unique characteristics, developing techniques for searching and mining blogs has become important. Several commercial websites like Blogpulse [5], Technorati [27], and Google Blog Search [25] are now available to mine and analyze blog content. These websites provide services such as searching blogs based on query keywords, ranking of blogs according to popularity, and identifying trends in keywords seen in the blogosphere. The research community has also begun to focus on blogs, and publications include studies on blog communities, and information diffusion through the blogosphere.

One important problem that has not yet been addressed in mining the blogosphere is to extract cohesive discussions from blog communities as time goes on, on any given subject or issue. In this work, we introduce this problem of story mining and propose a time-sensitive and community-sensitive model to group blog entries into stories, while leveraging distinct characteristics of blogs like entry time-stamps and community structure. We also propose a *Modified Time-sensitive Dirichlet Process* and use that in our model.

To group blog entries into stories, we use a probabilistic graphical model. The story is treated as a latent variable while the entry content, time-stamps, and inter-blog links are the observed variables. Our model does not require specifying the number of stories. The number of stories can be potentially infinite and is discovered by the model. Entries are first clustered by community and content. In each such cluster, the timestamps of entries are then used when grouping entries, so that newer entries are less likely to be grouped with older entries, since two entries with similar content but with wide separation in time are likely to be different stories. Once stories have been discovered, the most recent ones can be considered the *hot stories*.

Mining stories and hot stories from blogs has useful practical applications. It can help us see what issues are of interest to various domains and communities. Also, blog analytics are becoming an important tool for marketing research since blogs often contain frank unsolicited information about products. Discussions discovered through story mining can be processed to derive market intelligence [7].

CIKM'06, November 5–11, 2006, Arlington, Virginia, USA.
Copyright 2006 ACM 1-59593-433-3/06/0011 ...$5.00.

As an example, consider discussions on the blogosphere about the Apple iPod. Stories discussed would include the introduction of the iPod, and later that of the iPod Mini. In September of 2005, the hot story would have been the announcement of the iPod Nano. Or consider discussions about hurricanes in the Atlantic. Stories would include Hurricanes Dennis and Emily in July 2005, Katrina in August 2005, and in mid-September 2005, Hurricane Rita would have been the hot story. In this work, we wish to identify such stories and hot stories.

We summarize the contributions of this paper as follows:

1. We propose a two-stage model to cluster blog entries into stories and discover hot stories occurring in various communities in the blogosphere.

2. Our model provides a community-based clustering by incorporating links between blogs.

3. We propose a *Modified Time-sensitive Dirichlet Process* model and use it to incorporate time-stamps of blog entries while clustering them into stories.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 discusses the nature of the blogosphere, and how we approach the problem of discovering stories. Section 4 describes our proposed model. Section 5 presents experimental studies on real-world blog data and shows our model to be very effective in finding interesting communities and stories. We offer concluding remarks in Section 6.

2. RELATED WORK

We present related work here in four parts: 1) blog mining research, 2) use of probabilistic graphical models for content and relation analysis, 3) temporal mining, and 4) Web search result clustering.

2.1 Blog Mining

Since blogs are a new area of research, there has been relatively limited academic work in this area. The flow of information through blogs has been studied by Adar *et al.*[2] and Gruhl *et al.*[8]. Kumar *et al.*[15] studied the formation of communities, and their bursty evolution. Tseng *et al.* in [28] used tomographic clustering to form and visualize communities in the blogosphere. Both Kumar *et al.*[15] and Tseng *et al.*[28] used the link structure between blog entries to discover communities. In contrast, Ishida[11] discovered latent communities based on similarity in content of various blogs, without considering links.

To the best of our knowledge, there is no work in the literature that discovers stories while exploiting the unique characteristics of the blog medium.

2.2 Probabilistic Models for Content and Relation Analysis

Probabilistic models have become popular for text analysis. Often each topic is modeled as a probability distribution over words, and a document is viewed as a probabilistic mixture over these topics. The probability of generating word w_i in a given document is then given by:

$$P(w_i) = \sum_{j=1}^{T} P(w_i|z_i = j)P(z_i = j),$$

where z_i is a latent variable indicating the topic from which the w_i is drawn and T is the number of topics. $P(w_i|z_i = j)$ is the probability of w_i given topic j and $P(z_i = j)$ gives the prior probability of topic j.

In probabilistic Latent Semantic Analysis (PLSA) [9], topics are modeled as multinomial distributions over words, and documents are assumed to be generated from multiple topics. In Latent Dirichlet Analysis (LDA) [3], a distribution over topics is sampled from a Dirichlet distribution for each document. Each word is sampled from a multinomial distribution over words specific to the sampled topic. For D documents about T topics containing W words, we can represent $P(w|z)$ using a set of T multinomial distributions ϕ over the W words such that $P(w|z = j) = \phi_j^w$, and $P(z)$ with a set of D multinomial distributions θ over T topics, such that for a word in document d, $P(z = j) = \theta_j^d$.

Recently, the Author-Topic (AT) model [20] extended LDA to incorporate authorship information and the authors of [22] further extended the AT model to the Author-Recipient-Topic (ART) model, specifically for email, by regarding the sender-receiver pair as an additional variable.

Latent structure or groups have been discovered through pairwise relational data in the stochastic Blockstructures model [24], by modeling relations between entities probabilistically . A relation holds between a pair of entities with a probability that depends only on the group assignment of the entities. In [13], a non-parametric process was used to select the number of groups. The Group-Topic model [30] extends the Blockstructures model [24] to include attributes on the relations.

However, these models are not suitable for extracting stories from blogs, since they do not consider time and community structure along with the content, all of which are important characteristics of blogs. We propose a two-stage method that first groups blog entries into clusters corresponding to blogger communities and then extracts stories, using a combination of content and time stamps. Also, in our model, the number of stories need not be specified.

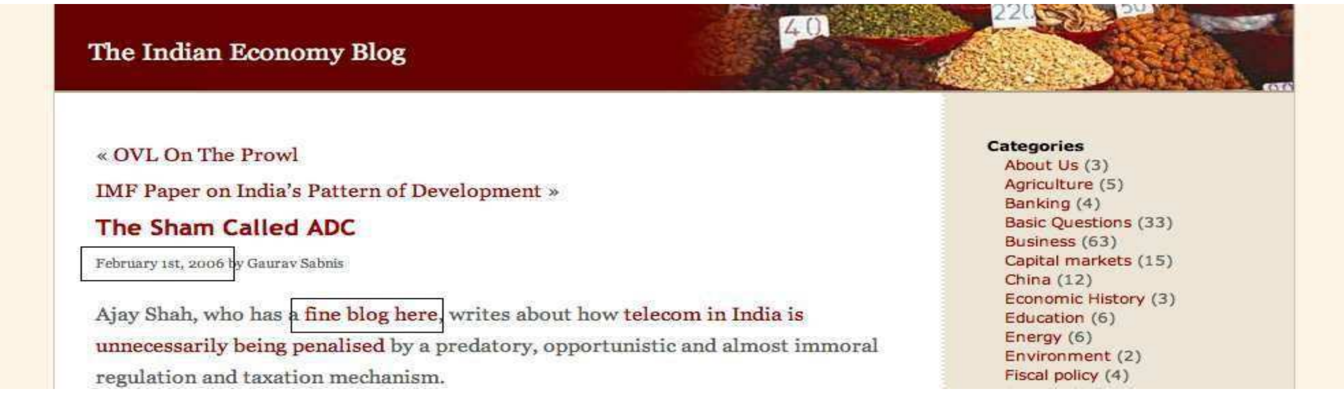
2.3 Temporal Mining

Recently, there has been some work on temporal mining for text streams. In [14], text streams are converted to temporal frequency data for discovering *burstiness* in the stream. The authors of [23] studied the evolution of themes (probability distributions over words) in a stream of documents. Other works, such as by Ma *et al.*[21], look for temporal trends in a text stream.

In contrast, our work is concerned with discovering coherent groups of documents (blog entries) while incorporating time. Additionally, the community structure is also used.

2.4 Web Search Result Clustering

Clustering methods have been used to group Web search results into groups to allow for better organization and easy navigation. Several clustering engines have been proposed and implemented. The Scatter/Gather system [6] was an early method. Grouper [31] performs clustering using a phrase-based algorithm called Suffix Tree Clustering. Vivisimo [29] is a commercial service that returns clustered Web search results, but uses an unpublished proprietary technique. Other systems include iBoogie [10], and Lingo [17]. The authors of [32] tackle the problem as a salient phrase ranking problem and use learning to discover the ranking, while Kumar *et*



The Indian Economy Blog

« OVL On The Prowl

IMF Paper on India's Pattern of Development »

The Sham Called ADC

February 1st, 2006 by Gaurav Sabnis

Ajay Shah, who has a fine blog here, writes about how telecom in India is unnecessarily being penalised by a predatory, opportunistic and almost immoral regulation and taxation mechanism.

Categories
About Us (3)
Agriculture (5)
Banking (4)
Basic Questions (33)
Business (63)
Capital markets (15)
China (12)
Economic History (3)
Education (6)
Energy (6)
Environment (2)
Fiscal policy (4)

Figure 1: An example of a blog entry. Note the timestamp, and the link to another blog in the entry.

al.[16] presented a graph theoretic approach to the problem. Blogs have unique characteristics, which makes it important to develop specialized techniques that can utilize these characteristics so as to create meaningful and useful stories.

3. MINING STORIES FROM THE BLOGOSPHERE

In this section, we discuss the motivation for mining stories from blogs, the issues involved, and our approach for addressing the problem.

3.1 Understanding the Blogosphere

The blogosphere is comprised of a large number of bloggers regularly updating their blogs with new entries. (Figure 1 shows a sample blog entry.) Since a blog usually corresponds to an individual who has certain topics of interest, the blog is likely to contain entries about those topics. Also, bloggers read other blogs about these topics and issues. Bloggers, in their entries, also link to and write about entries in these other blogs, thus creating a conversation in the blogosphere. These bloggers, with their shared interests, conversing over various topics and issues form online communities.

When mining the blogosphere for stories, the communiity structure can be very useful. A community in the blogosphere corresponds not only to a topic, but also to a particular viewpoint on the topic. For example, viewed based on content analysis alone, all blogs about politics may appear similar. However, since a political blogger is more likely to link to blogs that match his/her political viewpoint, the communities formed would correspond to a specific political ideology or viewpoint. This has been validated by [1]. Hence, uncovering and utilizing the community structure can be useful for better mining of coherent stories. As another example, consider discussions in the blogosphere about Apple Computer's move to the Intel platform. We may find technology-oriented bloggers who would view this news from a different perspective than perhaps bloggers interested in Apple's stock price. Again, the community structure can help differentiate these stories.

Before we proceed, we elaborate on our notion of a *story*. A story is comprised of a set of blog entries that are about a specific issue and that reflect a discussion in the blogosphere. Several stories may be retrieved from a blog database, for some query keywords.

Entries are usually triggered by online or offline events, and stories thus reflect events in the real world such as a news item, a new book or product, or a political or cultural event. Entries in a story have a time-stamp and stories have a time duration. Time can thus be useful in differentiating stories. Two blog entries that have similar content but which were written far apart in time were likely triggered by different events and thus should be part of different stories. For instance, two entries about the launch of the iPod written far apart in time probably deal with different models of the iPod. Thus it is important to take time-stamps of entries into consideration when clustering entries into stories.

3.2 Discovering Stories

Based on the above understanding of the blogosphere, we have developed a two-stage Content-Community-Time (CCT) story extraction model to discover stories from a blog database, given some query keywords. Entries relevant to the keywords are retrieved, and then grouped into stories based on their content, time-stamps, and community structure. Here, we give only an overview of our approach, but we go into a detailed description in Section 4.

3.2.1 Uncovering Community Structure

The community structure is first uncovered as follows. We construct a *Community Graph* where nodes correspond to the set of entries. We consider a blog entry A to have a link to a blog entry B if the blogger of A had hyperlinked to any entry in the blog that B belongs to, within the T days before posting entry A. We do this because this reveals that A's blogger had an interest in B's blog at the time of writing entry A. (We assume that the interests of a blogger remain consistent for T days). In this work, we set T to be 30 days. Also, if two entries come from the same blog, we consider them to be linked. We consider links to be directionless since we are interested in community structure, not the flow of information.

3.2.2 Grouping Entries

In the first stage of the CCT model, we consider content and community structure, and perform a coarse clustering. The set of retrieved entries is divided into *Community-Topic clusters* (CT-clusters) such that the entries in one cluster are from a group of bloggers that are likely to have a similar interest in the concerned topics, and who may discuss these issues online. (A related notion of *Topic-Communities* was introduced in [26] in the context of email.) In this model, the number of CT-clusters is determined based on the data, and does not have to be specified. This is very useful since the true number of CT-clusters for any query is unknown.

The second stage of the model discovers stories in each CT-cluster. Here content is considered again, but so as to create a finer clustering. Also entry time-stamps are used to assist clustering. Again, as for CT-clusters, the number of stories does not need to be specified.

4. CONTENT-COMMUNITY-TIME STORY EXTRACTION

In this section, we detail our *Content-Community-Time (CCT)* model and techniques for mining stories from blogs. In Section 4.1, we define some terms and set up notation. Section 4.2 describes the preprocessing performed. Sections 4.3, 4.4, and 4.5 then detail the process of extracting stories based on some keywords.

4.1 Preliminaries

We formally define here some terms and notions that we use throughout this paper.

Definition 1. (Blog): A *Blog* is a Webpage comprising of a sequence of dated *entries*, which has an associated *RSS* (Really Simple Syndication) feed. A blog is usually written by an individual who is called the *blogger*.

Definition 2. (Community Graph): A *Community Graph* is a graph with nodes corresponding to a set of entries, and edges depicting an interest relationship between the bloggers of the two entries.

Definition 3. (Community-Topic Cluster): A *Community Topic Cluster* (CT-cluster) is a set of blog entries generated by a community of bloggers that write about a topic of mutual interest, such as about politics, technology, or pop music, and link to each other's entries.

Definition 4. (Story):: A *Story* is a set of blog entries that are about a specific issue and that reflect a discussion in blogspace between members of an online community.

A blog database consists of blog entries e_i. Each entry e_i has associated with it a blog $b(e_i)$, and a time-stamp $t(e_i)$, and a document $d(e_i)$. The time-stamp $t(e_i)$ is the time at which the document $d(e_i)$ was posted at blog $b(e_i)$. Also associated with each entry is a set of hyperlinks embedded in the document.

When extracting stories, let n be the number of entries retrieved from the blog database for grouping into stories. For a set of n blog entries, we construct a Community-Graph G that contains n nodes, each corresponding to an entry, and each edge g_{ij} denotes a link between entry e_i and entry e_j.

The Community-Topic cluster that an entry e_i belongs to is denoted by $z(e_i)$, while the story it belongs to is denoted by $s(e_i)$. We use z to denote an n-dimensional vector containing $z(e_i)$ for each of n entries e_i. Similarly, s is a vector of stories $s(e_i)$ for each e_i. Also, z_{-i} and s_{-i} denote z and s excluding $z(e_i)$ and $s(e_i)$ respectively. We use n_j to denote the number of entries in a CT-cluster or story j.

Let the number of unique words be V. The words in a CT-cluster i are given by a multinomial distribution ϕ_i over the V words, such that the probability of seeing a word w in cluster j is $\phi_j(v)$. Similarly, θ_j represents a multinomial distribution over V words for a story j.

There are some parameters for the model. We use α_z and α_s to denote the *concentration parameters* for clustering into CT-clusters and stories, respectively. Hyperparameters γ_z and γ_s control the multinomial distributions ϕ and θ, while hyperparameter β controls the distribution of links between clusters. These parameters will be explained along with the model.

Symbol	Meaning
e_i	a blog entry
$t(e_i)$	timestamp of e_i
$d(e_i)$	content of e_i
$b(e_i)$	blog of e_i
$z(e_i)$	CT cluster that e_i belongs to
$s(e_i)$	story that e_i belongs to
z	n dimensional vector of $z(e_i)$ for all retrieved e_i
s	n dimensional vector of $s(e_i)$ for all retrieved e_i
z_{-i}	vector of CT clusters for all entries except e_i
s_{-i}	vector of storys for all entries except e_i
n	number of blog entries retrieved
n_j	number of blog entries in CT cluster or story j
G	link graph for n entries
g_{ij}	link between entry e_i and e_j in G
V	size of the vocabulary
ϕ_i	multinomial dist. over words for CT cluster i
θ_j	multinomial dist. over words for story j
α_z	concentration param. for CT clustering
α_s	concentration param. for story clustering
γ_z	hyperparam for multinomial dist. ϕ
γ_s	hyperparam for multinomial dist. θ
β	hyperparam for dist. over links across clusters

Table 1: Notation

4.2 Preprocessing

Entries from blogs are crawled from the Web and stored locally. The content of each entry is then processed to remove stop words, and stemming is performed using the Porter stemmer.

Then, to allow retrieval of entries based on query keywords, an inverted index is created and the contents of all the entries are inserted into this inverted index. In this work, we used the Apache Lucene [19] library, a standard open-source text search-engine for indexing the text. Along with the content, the entry date, the url, and the corresponding blog url are all stored in the index.

To allow construction of the Community Graph, for each entry we create a list of blogs hyperlinked in the corresponding blog within the T days before the entry itself. This *linked-blogs* list is also stored in the index so it can be retrieved along with the entries.

4.3 Retrieving Entries and Constructing Community Graph

The process of extracting stories begins with the user providing some query keywords. Based on these keywords, the Lucene index is queried to retrieve the top n relevant entries. The entry date, associated blog url, and the linked-blogs list.

We now construct the Community Graph, which represents the community structure. The graph G has n nodes, one for each entry. An edge g_{ij} exists between nodes e_i and e_j if $b(e_j)$ exists in the linked-blogs list of e_i or $b(e_i)$ exists in the linked-blogs list of e_j. The constructed graph is directionless. The Community Graph is used to incorporate community structure while finding CT-clusters.

4.4 Creating Community-Topic Clusters

We next use the set of retrieved entries, their content, and the associated community structure (represented by the graph) to group the entries into Community-Topic clusters.

A probabilistic model is used to do the clustering. Given entry documents $d(e_1), ..., d(e_n)$ and the Community Graph G, we cluster the entries. The cluster of an entry e_i is denoted by the latent variable $z(e_i)$. Gibbs sampling is used to compute the cluster labels based on an inference equation. For the inference, we need to find the probability that an entry e_i belongs to cluster j, given cluster memberships for all entries except e_i, the contents of all entries including e_i, and the graph G.

This can be broken down into three terms. The first term sets the prior probability for $z(e_i)$ based only on the cluster memberships of all other entries. The second term is the probability of generating the document $d(e_i)$ from cluster j based on the other documents in cluster j. The third term gives the probability of graph G being generated given the cluster memberships of all entries. We can write the inference equation as:

$$P(z(e_i) = j | z_{-i}, d(e_1), ..., d(e_n), G) \propto P(z(e_i) = j | z_{-i}) \\ P(d(e_i)|d(e_k) \forall z(e_k) = j) P(G|z), \quad (1)$$

where z is the vector of cluster assignments for all entries, and z_{-i} is the vector of cluster assignments for all entries except e_i.

We discuss in the following sections how each of the terms in the above equation are computed.

4.4.1 Chinese Restaurant Process

The first term on the right hand side of equation 1 gives the probability of an entry e_i being assigned to cluster j given the cluster assignments of all other entries. We set this probability based on a *Chinese Restaurant Process* (CRP). A Chinese Restaurant Process is a clustering mechanism that allows the number of clusters to be potentially infinite. This is quite useful for our purposes since the true number of CT-clusters is unknown and hard to preset.

In a Chinese Restaurant Process (CRP), the probability of a new object belonging to an existing group is directly proportional to the number of objects of that group seen previously. Also, the object may belong to a new group with some probability which is controlled by a *concentration parameter*. In this way, the number of groups formed is selected automatically by the model, and may grow with the number of objects. Using the CRP for our scenario, we can write

$$P(z(e_i) = j | z(e_1), ..., z(e_{i-1})) = \quad (2)$$
$$\begin{cases} \frac{n_j}{i-1+\alpha_z} \text{if j exists,} \\ \frac{\alpha_z}{i-1+\alpha_z} \text{if j is a new group,} \end{cases}$$

where n_j is the number of entries already assigned to cluster j, and α_z is the *concentration parameter*.

Note that in the above equation, the probability of entry e_i being in cluster j is conditioned on only the entries seen before it. However, it is known that CRP is exchangeable, meaning that the indices of e_i can be permuted without affecting the probabilities of z, and so we can treat e_i as the last object to be drawn from the CRP.

Thus, we can get $P(z(e_i) = j | z_{-i})$, the first term in equation 1 by assuming that all other entries have already been seen.

4.4.2 Entry Content

The second term in equation 1 incorporates the entry content by giving the probability of the content in entry e_i being generated from cluster j, given the content of all other entries in cluster j.

Each entry document $d(e_i)$ is represented by a bag of words from the vocabulary containing V words. We assume that each CT-cluster j is represented by a multinomial distribution ϕ_j over the vocabulary. The probability for entry $d(e_i)$ being generated from cluster j is then

$$P(d(e_i)|\phi_j) = \prod_{v \in vocabulary} \phi_j(v)^{e_i(v)}, \quad (3)$$

where $e_i(v)$ is the count of word v in the content of entry e_i, and $\phi_j(v)$ gives the probability that word v is generated from cluster j. We can avoid sampling ϕ by integrating it out, and using a Dirichlet multinomial to represent clusters.

$$P(d(e_i)|d(e_k) \forall z(e_k) = j) = \\ \int p(d(e_i)|\phi) p(\phi | d(e_k) \forall z(e_k) = j) d\phi, \quad (4)$$

where $p(\phi | d(e_k) \forall z(e_k) = j)$ is a posterior Dirichlet distribution which is derived from a base prior Dirichlet distribution. It can be shown that equation 4 can be written as:

$$P(d(e_i)|d(e_k) \forall z(e_k) = j) = \\ \frac{\Gamma(\sum_v f_v + \gamma_z) \prod_v \Gamma(e_i(v) + f_v + \gamma_z/V)}{\prod_v \Gamma(f_v + \gamma_z/V) \Gamma(\sum_v e_i(v) + \sum_v f_v + \gamma_z)}, \quad (5)$$

where f_v is the count of word v in $d(e_k) \forall z(e_k) = j$, $\Gamma()$ is the gamma function and γ_z is a hyperparameter.

4.4.3 Community Structure

The third term in equation 1 is designed to incorporate the blogger community structure such that intra-cluster linking is much less frequent than inter-cluster linking. We assume that each edge in G is generated independently, and the probability that there exists a link g_{ij}, depends on $z(e_i)$ and $z(e_j)$, the CT-clusters corresponding to e_i and e_j.

Given the vector of cluster assignments z, the probability of generating the graph G, given all cluster memberships is given by

$$p(G|z, \eta) = \prod_{a,b} (\eta_{ab})^{m_{ab}} (1 - \eta_{ab})^{m'_{ab}}, \quad (6)$$

where a and b range over all clusters, η_{ab} is the probability that there is an edge between an entry in cluster a and an entry in cluster b, m_{ab} is the number of edges between entries in a and b, and m'_{ab} is the number of pairs of entries in clusters a and b respectively that do not have a link between them. We can integrate out η in equation 7 using a symmetric Beta prior over every η_{ab}. Then

$$p(G|z) = \prod_{a,b} \frac{Beta(m_{ab} + \beta, m'_{ab} + \beta)}{Beta(\beta, \beta)}, \quad (7)$$

where β is a hyperparameter that controls the Beta prior.

4.4.4 Inference

Now we can substitute equations 2, 5, and 7 into equation 1 and infer the cluster memberships $z(e_i)$ for any entry given the cluster memberships of all other entries. We initialize clustering by arbitarily assigning a cluster to each entry, say by putting all entries in the same cluster.

A single Gibbs sampling iteration consists of looping through $i = 1, ..., n$ and sampling $z(e_i)$, in turn, using equation 1, based on the current cluster memberships of all other entries. Then for each entry, the cluster with the highest probability is assigned as the new cluster for the entry. The Gibbs sampling iteration is then repeated using the new cluster memberships. After running several iterations of the Gibbs sampler, we get cluster memberships $z(e_i)$ for each entry e_i.

4.5 Extracting Stories

In the next stage, we consider the entry content and time to further divide each CT-cluster into stories. As in the last stage, a probabilistic model is used to do the clustering. Given entry documents $d(e_1), ..., d(e_n)$ and their corresponding timestamps $t(e_i)$, we cluster the entries into stories. The story to which an entry e_i belongs is denoted by the latent variable $s(e_i)$. Gibbs sampling is used to compute the story labels based on an inference equation that incorporates time and content.

We can find the probability that an entry e_i belongs to story j, given story memberships for all entries except e_i, the contents of all entries including e_i, and the time-stamps of all entries. This can be broken down into two terms. The first term sets the prior probability for $s(e_i)$ based only on the story memberships of all other entries, and the time-stamps of all entries. The second term is the probability of generating the document $d(e_i)$ from story j based on the other documents in cluster j. We can write the inference equation as follows.

$$P(s(e_i) = j|s_{-i}, d(e_1), ..., d(e_n), t) \propto P(s(e_i) = j|s_{-i}, t) \\ P(s(e_i)|d(e_k)\forall s(e_k) = j), \quad (8)$$

where s_{-i} is the vector of cluster assignments for all entries except e_i, and t is the vector of time-stamps for all entries.

4.5.1 Time-Sensitive Clustering

The first term on the right hand side of equation 8 gives the probability of an entry e_i being assigned to story j given the story assignments of all other entries, and the time-stamps of all entries.

For this, we propose a *Modified Time-sensitive Dirichlet Process* model (based on [12]), where the probability of entry e_i belonging to story j given the history of story assignent $s(e_1), ..., s(e_{i-1})$, and the time-stamps $t(e_1), ..., t(e_i)$ is given by

$$P(s(e_i) = j|s(e_1), ..., s(e_{i-1}), t(e_1), ..., t(e_i)) = \quad (9)$$
$$\begin{cases} \frac{w(t(e_i),j)}{\sum_{j'} w(t(e_i),j')*(1+\alpha_s)} \text{if j exists,} \\ \frac{\sum_{j'} w(t(e_i),j')*\alpha_s}{\sum_{j'} w(e(t_i),j')*(1+\alpha_s)} \text{if j is a new group,} \end{cases}$$

where α_s is a concentration parameter, and the function $w(t, j)$ gives the weight of a story j at time t. We define function $w(t, j)$ below.

A Time-sensitive Dirichlet Process is similar to the Chinese Restaurant Process, but also incorporates time while assigning clusters. For a new object, the probability of joining an existing group depends on the members of the groups and their age, while it may create a new group with some probability (controlled by the concentration parameter). The *influence* of existing groups decays with the age of members in the group. However, this model is not exchangeable, and we cannot treat any object as the last object to be seen. The first term in equation 8, $P(s(e_i = j)|s_{-i})$ is then given by

$$P(s(e_i) = j|s_{-i}, t) \\ \propto P(s(e_i) = j|s(e_1), ..., s(e_{i-1}), t(e_1), ..., t(e_i)) \\ \prod_{m=i+1}^{n} P(s(e_m)|s(e_1), ..., s(e_{m-1}), t(e_1), ..., t(e_m)). \quad (10)$$

In prior work, the authors of [12] proposed a Time-sensitive Dirichlet Process model where the probability of entry e_i belonging to story j given the history, would be given by

$$P(s(e_i) = j|s(e_1), ..., s(e_{i-1}), t(e_1), ..., t(e_i)) = \quad (11)$$
$$\begin{cases} \frac{w(t(e_i),j)}{\sum_{j'} w(t(e_i),j')+\alpha_s} \text{if j exists.} \\ \frac{\alpha_s}{\sum_{j'} w(e(t_i),j')+\alpha_s} \text{if j is a new group.} \end{cases}$$

However this model has a limitation. The first term in the denominator in the RHS of equation 11, varies significantly since it is very data dependent. The probability of a new cluster is then very different for each term in equation 10, but we would like the possibility of creating new clusters to be uniform. Thus, we proposed equation 9 where probability of being assigned a new cluster is same for all terms.

Now, the weight function $w(t, j)$ used above is defined such that the the influence of a story decays with the age of its member entries. It is given by

$$w(t, j) = \sum_{\{i|t(e_i)<t, s(e_i)=j\}} k(t - t(e_i)), \quad (12)$$

where $k(t)$ is a kernel that controls the decay of influence influence of an cluster. The weight decays with time and so a new entry is less likely to join a much older story. We can use a kernel that causes exponential decay of the influence of an entry [12], as follows:

$$k(t) = \begin{cases} exp(-\lambda t), \text{ if } t \geq 0, \\ 0, \text{ if } t < 0, \end{cases}$$

where λ is a decay parameter.

4.5.2 Entry Content

The second term in equation 8 gives the probability of seeing the content in entry e_i being generated from story j, given the content of all other entries in story j. Similar to the last stage, we can write

$$P(d(e_i)|d(e_k)\forall s(e_k) = j) = \\ \frac{\Gamma(\sum_v f_v + \gamma_s) \prod_v \Gamma(e_i(v) + f_v + \gamma_s/V)}{\prod_v \Gamma(f_v + \gamma_s/V)\Gamma(\sum_v e_i(v) + \sum_v f_v + \gamma_s)}, \quad (13)$$

where f_v is the count of phrase v in $d(e_k) \forall s(e_k) = j$, $\Gamma()$ is the gamma function and γ_s is a hyperparameter.

4.5.3 Inference

Now we can substitute equations 10 and 13 into equation 8 and infer the story $s(e_i)$ for all entries using Gibbs sampling as performed in Section 4.4.4. After running several iterations of the Gibbs sampler, we get story assignments $s(e_i)$ for each entry e_i.

4.5.4 Selecting Stories and Hot Stories

At the end of this stage, for each entry in each CT-cluster, we have assigned a story label. We are interested in extracting interesting stories from the blogosphere. The importance of a story can be in some sense measured by the number of entries in it. For instance, many stories might have just a single entry. Such a story is unlikely to reflect an important event or happening, and is likely to be a personal musing on the subject by a blogger. We thus select the stories that contain several entries.

Since each entry has an associated time-stamp, the time and duration of the story can be determined. Once we have extracted the stories, we can find **hot stories** by seeking those that are recent.

5. EXPERIMENTS

We implemented the proposed CCT model and used it for mining stories from the blogosphere. For this purpose, we crawled the Web and collected over a million blog entries.

We used the model for several query keywords and studied the results. Since no ground truth is available for these blogs, and the relevance and quality of results may be subjective, it is not practically feasible to give a quantitative evaluation of results. Instead, we demonstrate the effectiveness of our model and its various aspects through several case studies.

In the case studies, we show stories discovered using the proposed model, describe the Community-Topic clusters formed, and how they correspond to actual blogger communities. Also we show story extraction differs when performed using the following two alternative models.

Time-Insensitive Story Extraction

To understand the utility of timestamps in extracting stories, we experimented with a model that replaces the Time-sensitive Dirichlet process in the second stage of the CCT model with a Chinese Restaurant Process. This makes the story clustering insensitive to time.

Our testbed is described in Section 5.1, selection of model parameters is discussed in Section 5.2, and several case studies are presented in Section 5.3.

One-Stage Story Extraction

It is possible to perform the story extraction by combining community structure, content, and time, for clustering together in one stage. The inference equation for such clustering would be

$$P(s(e_i) = j | s_{-i}, d(e_1), ..., d(e_n), G) \propto P(s(e_i) = j | s_{-i}) \\ P(d(e_i) | d(e_k) \forall s(e_k) = j) P(G|s), \tag{14}$$

where the first term would be computed using the Time-sensitive Dirichlet process as in equation 10, the second term would be given by equation 13, and the third term would be given by equation 7. The inference can then proceed by Gibbs sampling as in Section 4.4.4.

5.1 Testbed

We prepared a blog database for use as a testbed for our studies. The blog database contained over 1 million entries along with timestamps, the corresponding blog url, and embedded hyperlinks to other blogs.

5.1.1 Crawling the Blogosphere

The blog database was constructed by crawling the blogosphere for over 10 months, from December 2004 to September 2005. The crawling was started from a small set of handpicked blogs, and continued along the out-links from these blogs. For our purpose, blogs are webpages that have an associated RSS stream. For over $2,000$ blogs, new entries were retrieved on a daily basis. Also, we stored all blog entries linked to from any of these $2,000$ blogs.

5.1.2 Entry Time-stamps

Time-stamps for blog entries may be available from the corresponding RSS feed. However, for many blogs, the time-stamp is missing or malformed in the RSS feed. For these, we recovered the time-stamps by examining the HTML files or the URL. Several hand-crafted regular expressions were used to extract dates in a variety of date formats, and some XPath expressions were used to look for dates in certain locations in the documents. For this work, we consider time at the granularity of dates.

5.2 Model Parameters

In the CCT model, there are six parameters that control the various factors in finding community-topics and queries. These parameters are:

γ_z: Hyperparam for Dirichlet distribution for CT clusters

γ_s: Hyperparam for Dirichlet distribution for stories

α_z: Concentration param for Chinese Restaurant Process

α_s: Concentration param for Time-Sesitive Dirichlet Model

β: Hyperparam for Beta prior for community structure

λ: Entry weight decay param

These parameters must be tuned in accordance with the our objectives. For instance, γ_z for the first stage must be such that the clustering is coarse, since at this stage we want to find only the broad community-topics and not the individual stories. In contrast, for discovering stories, γ_s should be such that finer clustering is done so as to find the individual stories. The concentration parameters α_z and α_s should be such that there is a small probability of an entry being assigned to a new cluster. The decay parameter λ must be set to appropriately control the decay of weight of an entry decays with time. The effect of community structure on clustering is controlled by β.

We tuned these parameters based on a user study, where the community-topics and stories obtained with various parameters were examined manually to evaluate the quality of results. The best parameters were thus chosen and used.

For the hyperparameters γ_z and γ_s, we used the values 0.1 and 50 respectively. The concentration parameters α_z and α_s were chosen to be 2 and 0.01 respectively. The value for parameter β was chosen to be 10, and that for the decay parameters λ was taken to be 0.5.

For the Time-Insensitive story-extraction model, we used the same parameters, except that the concentration parameter for CRP in the second stage was set to be 2. When experimenting with the One-Stage story extraction model, we set γ_s to be 50, α_s, to be 0.01, λ to be 0.5, and β to be 10.

5.3 Case Studies

In this section, we present three case studies of story mining using the proposed CCT model. The query keywords used for the three case studies are "Tsunami", "India China", and "Sony". For each, we use the top 200 blog entries for grouping into Community-Topic clusters and stories.

The three case studies demonstrate different aspects of our story-mining model.

1. The "Tsunami" query (Section 5.3.1) shows CT-clusters and stories relating to the tsunami disaster of 2005 in the Indian Ocean, and the discovered hot stories. Also, we compare this with the Time-Insensitive story extraction model.

2. The "India China" case study (Section 5.3.2) demonstrates discovery of communities, and of stories in communities using the CCT model. Then it shows a comparison with the One-Stage story-extraction model.

3. The "Sony" case study (Section 5.3.3) shows stories and hot stories mined about the company Sony and its products, using the CCT model.

5.3.1 Tsunami

For the "Tsunami" keyword, we found two CT-clusters. We represent them in Table 2 by the top 5 most frequent words in each cluster (excluding the query word itself).

A closer inspection of the content of the entries in each CT-cluster show that the CT-cluster 1 contains entries that talk about the tsunami disaster, while entries in CT-cluster 2 provide videos and images relating to the tsunami. CT-cluster 2 contains few entries but gets separated into a separate cluster since it is distinct from other discussions about the tsunami.

We now look at CT-cluster 1 in greater detail. In the story extraction phase, several stories were extracted from this cluster. We examined a few stories. The top 5 most frequent words for these stories are listed in Table 3.

We examine manually the content of entries in each of these stories.

Story 1 contains entries about the Katrina hurricane that compare the devastation caused to that of the South Asian tsunami. The entries are dated around late August and early September 2005. A few days after the Katrina hurricane, the enormity of the disaster became apparent, and there were frequent comparisons with the tsunami of a few months ago.

Story 2 contains entries dated late December 2004 through February 2005 that talk about the tsunami soon after it happened. Entries talk about the disaster, and about donation and relief efforts.

Story 3 occurred in July 2005. An earthquake near the Nicobar island in the Indian Ocean, the same region where the December 2004 tsunami originated, caused fears of another tsunami, and a tsunami warning was announced. These entries discuss this event. The content of these entries is actually similar to the content of the entries about the tsunami, but they get separated into different stories because of the influence of time-stamps.

CT	Most Frequent Words
1	katrina, donate, news, relief, hurricane
2	video, new, list, bittorrent, image

Table 2: "Tsunami" CT-clusters

	Most Frequent Words	Time
1	katrina, hurricane, donate, relief, asian	Aug-Sep
2	news, donation, earhquake, relief, disaster	Dec-Feb
3	earthquake, island, report, photo, nicobar	Jul
4	help, relief, donate, countries, debt	Jan
5	sarvodaya, asia, million, raised, disaster	Aug-Sep

Table 3: Stories in "Tsunami" CT-clusters 1

Entries in Story 4, which talk about donations to the tsunami relief efforts, are dated in early January. These entries, however, do not discuss the devastation caused and the details of the tsunami itself, and are hence distinct from Story 2.

Story 5 contains entries dated around August/September 2005 which look at the tsunami relief efforts months after the tsunami. The donations raised and the success of certain relief programs are discussed.

In CT-cluster 2, each of the entries gets assigned to a different story. An examination of their content shows that though they are similar at a coarse level, the distinctness of the entries means that each of the entries corresponds to a different story.

	Most Frequent Words	Time
1	katrina, donate, hurricane, american, relief	Jan-Sep
2	lanka, sarvodaya, affect, live, donate	Feb-Aug
3	face, victim, india, pirate, press	Mar-Aug

Table 4: Stories in "Tsunami" CT-clusters 1 **when timestamps not used**

CT	Most Frequent Words
1	state, country, economy, news, service
2	country, develop, news, market, companies
3	business, world, service, climate, market

Table 5: "India China" CT-clusters

http://www.desipundit.com/
http://www.rediff.com/rss/
http://dipforum.civiblog.org/blog
http://deesha.org/
http://www.dotnetindia.com

Table 6: Blogs in "India China" CT-cluster 1

Hot Story Detection

From the stories discovered above, we can discover the hot stories. At the time this dataset was collected and stored, September 2005, the hot stories related to the tsunami were 1) comparisons with the Katrina hurricane disaster, and 2) an evaluation of tsunami relief efforts.

http://feeds.feedburner.com/indiastockblog
http://feeds.feedburner.com/chinastockblog
http://indianeconomy.org
http://simonworld.mu.nu/
http://feeds.feedburner.com/sramanamitra

Table 7: Blogs in "India China" CT-cluster 2

	Most Frequent Words	Time
1	time, story, rise, businessweek, tibet	Aug
2	industrial, elephant, chinese, growth, million	Jul-Sep

Table 8: Stories in "India China" CT-cluster 1

	Most Frequent Words	Time
1	service, uranium, years, climate, companies, global	Jul-Aug
2	business, economy, outsource, service, infosys	Aug
3	businessweek, rise, chinese, economy, service	Aug

Table 9: Stories in "India China" CT-cluster 3

Time-Insensitive Story Mining

Table 4 shows some stories discovered using the Time-Insensitive story-mining model. It is notable here that the stories are rather spread out in time. Stories 1 and 4 (from Table 3) can be seen combined into one story (Story 1), which is clearly undesirable.

Story Clustering Using Only Time

It is worth noting at this point that a clustering based solely on time-stamps and not on content (i.e. by discovering story boundaries along the time axis), would also not give satisfactory results. For instance, in this case study, Stories 1 and 5 occur during overlapping time durations. Examination of content is important for differentiating the two.

5.3.2 India China

We performed story mining with the query words "India China" in order to discover interesting stories relating to the two countries. We found three CT-clusters, which are represented in Table 5 by the top 5 most frequent words in each cluster (excluding the query words themselves).

We examine closely the Community-Topic clusters retrieved. Table 6 shows some of the blogs in the community corresponding to CT-cluster 1. Checking these blogs shows they indeed form a community. The authors of these blogs post entries relating to India, and regularly comment on and link to each other's entries. Similarly, the blogs in CT-cluster 2 (shown in Table 7) form a community of economics bloggers, who write about Indian and Chinese economies. The third CT-cluster contains blogs that are not connected to the first two communities.

Stories are extracted from these CT-clusters. Table 8 and Table 9 show some interesting stories seen in CT-clusters 1 and 3 (represented by the top 5 most frequent words).

In CT-cluster 1, the we see a story that involves bloggers discussing an article about India and China that appeared in *Business Week* magazine. A story about the *Business Week* article was also seen in CT-cluster 3. This demonstrates that different communities can discuss the same stories. A user can browse these stories about the same issue in different communities and get different perspectives.

	Most Frequent Words	Time
1	economiy, world, company, develop, nation	Jul-Sep
2	india, business, businessweek, special, issue	Aug
2	outsource, service, country, public, offshore	Aug

Table 10: Stories for "India China" with one-phase mining

CT	Most Frequent Words
1	ericsson, music, apple, image, endgadget
2	hdbeat, entry, pad, category, line
3	erricson, w550i, product, infosyncworld, manufacture

Table 11: "Sony" CT-clusters

Other stories seen in Tables 8 and 9 include a discussion of India's and China's energy needs, and disucssions on the growth in their industrial sector and services outsourcing businesses.

One-Stage Story Mining

In Table 10, we show some stories obtained using the One-Stage story-mining model for the "India China" query. We see here that the story related to the *Business Week* article appears only once, since similar stories from the two communities have been combined into one. The user thus loses the advantage of reading the story from two different perspectives.

Also, by not finding CT-clusters first, we would not have discovered the communities of bloggers that we found with the CCT model. Thus, the advantage of the CCT model is apparent.

5.3.3 Sony

For the "Sony" case study, the system discovered three CT-clusters. We represent them in Table 11 by the top 5 most frequent words in each cluster. Here CT-cluster 1 contains most of the entries retrieved. We will examine this cluster in greater detail below. On examining CT-cluster 2, we find that it contains corresponds to a community interested in High-Definition Television, and entries in this CT-cluster discuss Sony products from that perspective. CT-cluster 3 corresponds to the community around the Infosyncworld website.

We now examine a few of the stories in CT-cluster 1. These stories are shown in Table 12 represented by the top 5 most frequent words. Story 1 contains entries discussing the agreement between Sony and Apple Computers to sell songs at the iTunes music store (ITM). These entries are dated around the second week of September 2005 when this was announced. Story 2 contains entries that review the Sony Cyber Shot camera when a new version was released around late July, 2005. Story 3 contains entries that discuss the launch of the Sony Ericsson w800i phone in March 2005.

	Most Frequent Words	Time
1	music, apple, launch, japan, itm	Sep
2	review, cyber, shot, image, firmware	Jul
3	ericsson, launch, w800i, release, order	Mar

Table 12: Stories in "Sony" CT-clusters 1

Hot Story Detection

Here again, we can identify a hot story. In September 2005 (when the dataset was collected), Sony had just announced an agreement with Apple, which was the hot story relating to Sony at that time.

5.3.4 Discussion

The above case studies demonstrate that the CCT model can indeed mine stories and hot stories successfully. The stories found can be seen to correspond to real events, such as the Katrina hurricane or Sony-Apple tie-up. The CT-clusters found can be seen to correspond to actual online communities, like the community of Indian bloggers. We also saw that the same story may be detected in different communities, like we saw with the story about the *Business Week* article, thus allowing the user to see different perspectives on the story. Using a one-stage approach to story mining may allow finding stories, but we lose the ability to find the communities where these discussions are occurring, and to differentiate between them. Also, we saw that the use of time-stamps indeed does help to better find stories by differentiating better based on when they occurred.

6. CONCLUSION

Blogs have emerged as an important form of Web content, making it important to develop techniques to mine content from the blogosphere. Though similar to other Web content in some ways, blogs have unique characteristics which must be considered and exploited while analyzing information from the blogosphere.

In this work, we presented the Content-Community-Time Story Extraction model to extract stories from the blogs. The model incorporates the content of blog entries, their time-stamps, and the community structure to find stories. To study and evaluate our model, we implemented a system and used it for a database of over a million blog entries from the Web. We presented case studies to demonstrate that the proposed model is successful in finding interesting stories from blogs.

7. REFERENCES

[1] Lada A. Adamic and Natalie Glance. The political blogosphere and the 2004 u.s. election: Divided they blog. *Proceedings of KDD Workshop on Link Analysis and Group Detection LinkKDD*, 2005.

[2] E. Adar and L. A. Adamic. Tracking information epidemics in blogspace. In *Web Intelligence*, 2005.

[3] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal on Machine Learning Research*, 3:993–1022, 2003.

[4] Blogger. www.blogger.com.

[5] Blogpulse. www.blogpulse.com.

[6] Douglass Cutting, David Karger, Jan Pedersen, and John W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of 15th Annual International ACM SIGIR Conference on Information Retrieval*, 1992.

[7] Natalie Glance, Matthew Hurst, Kamal Nigam, Matthew Siegler, Robert Stockton, and Takashi Tomokiyo. Deriving market intelligence from online discussion. In *ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining*, 2005.

[8] D. Gruhl, R. V. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. *SIGKDD Explorations*, 6(2):43–52, December 2004.

[9] T. Hoffman. Probabalistic latent semantic analysis. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 1999.

[10] iBoogie. www.iboogie.com.

[11] K. Ishida. Extracting latent weblog communities: A partitioning algorithm for bipartite graphs. In *Proceedings of 2nd Annual Workshop on the Weblogging Ecosystem*, 2005.

[12] X. Jhu, Z. Ghahramani, and J. Lafferty. Time-sensitive dirichlet process mixture models. *Technical Report, CMU-CALD-05-104*, 2005.

[13] C. Kemp, T. L. Griffiths, and J. Tenenbaum. Discovering latent classes in relational data. *Technical Report, MIT CSAIL*, 2004.

[14] Jon Kleinberg. Bursty and heirarchical structure in streams. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.

[15] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. On the bursty evolution of blogspace. In *Proceedings of the12th International Conference on World Wide Web (WWW)*, pages 568–576, 2003.

[16] Ravi Kumar, Uma Mahadevan, and D. Sivakumar. A graph-theoretic approach to extract storylines from search results. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2005.

[17] S. law, O. Jerzy, and S. Dawid. Lingo: Search results clustering algorithm based on singular value decomposition, 2004.

[18] LiveJournal. www.livejournal.com.

[19] Apache Lucene. lucene.apache.org.

[20] M. Steyvers M. R.-Zvi, T. Griffiths and P. Smyth. The author-topic model for authors and documents. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 21, 2004.

[21] J. Ma and S. Perkins. Online novelty detection on temporal sequences. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.

[22] A. McCallum, A. Corrada-Emmanuel, and X. Wang. The author-recipient-topic model for topic and role discovery in social networks: Experiments with enron and academic email. *Technical Report UM-CS-2004-096*, 2004.

[23] Qiaozhu Mei and ChengXiang Zhai. Discovering evolutionary theme patterns from text - an exploration of temporal text mining. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2005.

[24] K. Nowicki and T. A. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 2001.

[25] Google Blog Search. blogsearch.google.com.

[26] Xiaodan Song, Ching-Yung Lin, Belle L. Tseng, and Ming-Ting Sun. Modeling and predicting personal information dissemination behavior. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2005.

[27] Technorati. www.technorati.com.

[28] B. L. Tseng, J. Tatemura, and Y. Wu. Tomographic clustering to visualize blog communities as mountain views. In *Proceedings of 2nd Annual Workshop on the Weblogging Ecosystem*, 2005.

[29] Vivisimo. www.vivisimo.com.

[30] X. Wang, N. Mohanty, and A. McCallum. Group and topic discovery from relations and text. In *Proceedings of KDD Workshop on Link Analysis and Group Detection (LinkKDD)*, 2005.

[31] Oren Zamir and Oren Etzioni. Grouper: a dynamic clustering interface to Web search results. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1361–1374, 1999.

[32] H. Zeng, Q. He, Z. Chen, W. Ma, and J. Ma. Learning to cluster web search results. In *Proceedings of 27th Annual ACM SIGIR*, 2004.