

Simultaneously Modeling Semantics and Structure of Threaded Discussions: A Sparse Coding Approach and Its Applications*

Chen Lin[†], Jiang-Ming Yang[‡], Rui Cai[‡], Xin-Jing Wang[‡], Wei Wang[†], Lei Zhang[‡]

[†]School of Computer Science, Fudan University. {chen_lin, weiwang1}@fudan.edu.cn

[‡]Microsoft Research, Asia. {jmyang, ruicai, xjwang, leizhang}@microsoft.com

ABSTRACT

The huge amount of knowledge in web communities has motivated the research interests in threaded discussions. The dynamic nature of threaded discussions poses lots of challenging problems for computer scientists. Although techniques such as semantic models and structural models have been shown to be useful in a number of areas, they are inefficient in understanding threaded discussions due to three reasons: (I) as most of users read existing messages before posting, posts in a discussion thread are temporally dependent on the previous ones; It causes the semantics and structure to be coupled with each other in threaded discussions; (II) in online discussion threads, there are a lot of junk posts which are useless and may disturb content analysis; and (III) it is very hard to judge the quality of a post. In this paper, we propose a sparse coding-based model named **SMSS** to **Simultaneously Model Semantics and Structure** of threaded discussions. The model projects each post into a topic space, and approximates each post by a linear combination of previous posts in the same discussion thread. Meanwhile, the model also imposes two sparse constraints to force a sparse post reconstruction in the topic space and a sparse post approximation from previous posts. The sparse properties effectively take into account the characteristics of threaded discussions. Towards the above three problems, we demonstrate the competency of our model in three applications: reconstructing reply structure of threaded discussions, identifying junk posts, and finding experts in a given board/sub-board in web communities. Experimental results show encouraging performance of the proposed **SMSS** model in all these applications.

Categories and Subject Descriptors

I.5.1 [Pattern Recognition]: Models - Statistical

General Terms

Algorithms, Experimentation

Keywords

Threaded discussion, sparse coding, reply reconstruction, junk identification, expert finding

*This work was done during the first author was an intern in Microsoft Research Asia.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGIR '09, July 19–23, 2009, Boston, Massachusetts, USA.

Copyright 2009 ACM 978-1-60558-483-6/09/07 ...\$5.00.

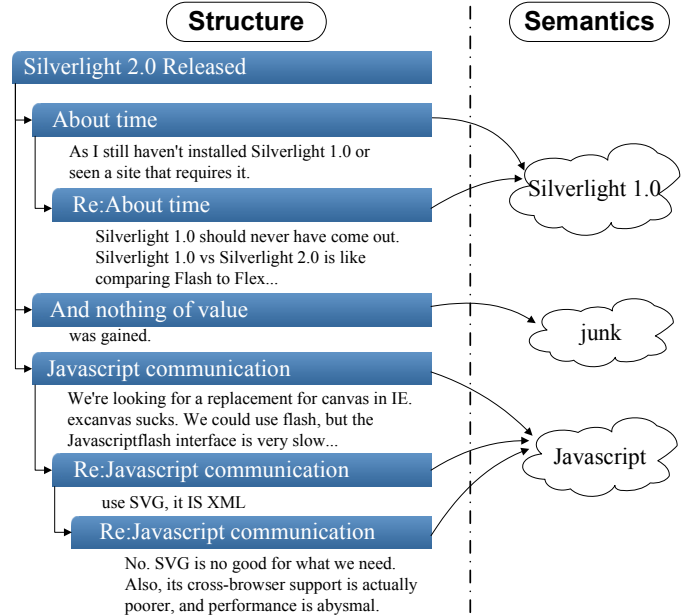


Figure 1: An example of the structure and semantics of a discussion thread from Slashdot.

1. INTRODUCTION

Threaded discussion has long been a popular option for web users to communicate with others, from thousands of web forum sites, mailing lists, chat rooms to web logs, instant messaging groups, and so on. As we are in the age of Web 2.0, threaded discussion acts as an important tool to facilitate collaborative content contributions. With millions of users' contribution, highly valuable knowledge and information have been accumulated on various topics. These topics include recreation, sports, games, computers, art, society, science, home, health, etc.¹; and especially some topics related to our daily life which are rarely seen in traditional web pages. As a result, recent years have witnessed more and more research efforts on mining information from online discussion threads.

A discussion thread usually originates from a root post by the thread starter. Fig. 1 gives an intuitive description of a discussion thread². It contains 7 posts. The first post is a piece of news about the release of "SilverLight 2.0". Some users comment on this post, *i.e.*, the 2nd and 3rd posts are about the "update time"; some users have further questions and initiate sub-discussions, *i.e.*, the 5th 6th and 7th posts

¹<http://directory.big-boards.com/>

²<http://developers.slashdot.org/comments.pl?sid=1000769>

are about “Javascript communication”; and others troll or complain in some posts, *i.e.*, the 4th post. As more users are joining in and making comments, the discussion thread grows, forming a nested dialogue *structure* as shown in the left part of Fig. 1. Furthermore, threaded discussions show rich complexity in the *semantics*. Since users always respond to others, previous posts affect later posts and cause the topic drift in a discussion thread, as shown in the right part of Fig. 1.

Mining discussion threads is challenging. The following reasons prevent people from fully exploring the value of discussion threads. (I) Posts in a discussion thread are temporally dependent upon each other. A newcomer may read some of the previous messages before posting. Replies indicate sharing of topics and vice versa. Hence a post is a mixture and mutation of previous posts by nature. Unfortunately, such specific orderings and intra dependencies of posts in one thread are neglected by most existing research methods. (II) As discussion threads are born to encourage content distribution and contribution, they sometimes become targets of spammers. Meanwhile, some messages are casual chitchat and are needless to analyze. Posts of either spam or chitchat are regarded as junk posts. Though chitchat might serve to foster relationships between participants, such kinds of junk posts are useless and may disturb content analysis in some applications, such as Question & Answer mining, expert finding, and so on. (III) It is very hard to estimate the quality of a post. It is true that valuable posts are usually long articles; but there is also a remarkable amount of long meaningless posts that are not meant to help others, and on the contrary there are some short insightful posts that inspire a lot of people. Thus measurements solely by content length or content relevance usually do not work.

Although previous research efforts have made progress in many information retrieval scenarios, few of them are suitable for mining online discussion threads. Current related work can be mainly classified into two categories: semantics-based or linking structure-based. Probabilistic topic models [9, 3] fall into the former class. Their main idea is to project documents to some latent topic space. However, most topic models assume documents in one collection to be exchangeable, *i.e.*, their probabilities are invariant to permutation. It is contrary to the reply relationships among posts. Other works such as HITS [13], PageRank [16] and their successive research belong to the latter class. These research efforts try to identify the importance of content of each document. They mainly utilize link relationships among various documents. Unfortunately, most community sites such as those forums powered by vBulletin and phpBB do not provide explicit reply relationships among posts. Furthermore, semantics and structure of a threaded discussion are highly dependent on each other. That is, when semantics evolves, the dialogue structure changes, and vice versa. This is the nature of discussion threads. Most previous research efforts can not solve this problem directly as they are solely from the semantic-centric view or from the structure-centric view.

In this paper, towards simultaneously modeling semantics and structure of threaded discussions, we propose a novel sparse coding-based approach called **SMSS**, which has the ability to integrate both the semantic and structure aspects of threaded discussions. For modeling semantics, similar to existing topic models, our approach can discover latent topics as bases to capture semantics in discussion threads; and each post is represented as a fusion of these semantic bases. Considering that users usually only discuss one or two top-

ics in one individual post, we also add a sparse constraint in this fusion. For modeling structure, to embed the potential reply structure, topics of each post can just be sampled from the topics of those earlier posts in the same thread. We also add a sparse constraint here because one post only tries to reply one or two previous posts. At last, to demonstrate the effectiveness of the proposed **SMSS** model and to investigate the above problems (I)~(III), we evaluate our model with the following three applications, respectively:

- **REPLY RELATIONSHIP RECONSTRUCT.** We reconstruct the reply relationship among posts based on both their semantic and structure coefficients estimated by **SMSS**. Experimental results demonstrate that **SMSS** achieves higher precision than previous topic models.
- **JUNK DETECTION.** We illustrate how **SMSS** is competitive on junk detection. Concretely, we introduce a background topic into the topic space and detect the junk posts based on their coefficients on the background topic. Experimental results show that **SMSS** outperforms classification-based algorithms and existing topic models.
- **EXPERT FINDING.** We try to find out experts in a given board/sub-board. Since the key problem of expert finding is to estimate the author’s average post quality, we demonstrate the value of structure information from **SMSS** and the capability of measuring post content quality.

This paper is organized as follows. Section 2 briefly reviews related work; Section 3 presents the details of the proposed **SMSS** model; and Section 4 describes how the **SMSS** model are leveraged in applications. Experimental results are shown in Section 5. In Section 6, we draw the conclusion and plan the future work.

2. RELATED WORK

In this section, we introduce previous research efforts which are related to our work. We begin with the methodologies, briefly summarizing their innovations, advantages, and drawbacks, and then we give an overview of recent technical trends in mining threaded discussions.

2.1 Semantic Models

Much work has been proposed to model text semantics (or topics), such as latent Dirichlet allocation (LDA) [3] and the more general discrete component analysis [4]. They decompose documents into a small number of topics which are distributions over words. In LDA, each document is produced by choosing a distribution over topics with a Dirichlet prior; each word is sampled from a multinomial of topic-word association. Some work has been proposed to extend LDA to model multiple relationships, such as authorship [18], email [14], and etc.; while others tried to model the background, topic, and document specific words simultaneously [5]. Some recent publications began to model time dependency among documents, e.g., [2] tried to model the dependency in discrete time periods and [21] considered time to be continuous. However, these models only considered the topic drift within two adjacent time periods, which is not suitable for the hierarchical intra dependency of posts in one discussion thread. In general, the main drawback of semantic models for discussion thread analysis is that they only capture the semantic information but ignore the temporal structural information.

2.2 Structural Models

There have also been studies focusing on structural modeling. A collection of documents and linkages between them are constructed as a graph. The nodal importance or nodal quality can be estimated by the structural centrality of the nodes in the graph, where the importance refers to authority, popularity, expertise or impact in various applications. In [11], several merits are presented to measure the nodal structural centrality. Classical structural models include HITS [13] and PageRank [16]. HITS is carried out in an iterative manner, propagating the authority and hub of one node to another. PageRank simulates the random walks of a surfer, who continuously jumps from one web-page to another linked page with a uniform probability, with a damping factor. Although structural models have been applied with remarkable success in different domains, it is not suitable to analyze threaded discussions. This is because these works highly rely on the link structure among documents; while there is usually no explicit link structure among the posts in a discussion thread.

2.3 Mining Threaded Discussions

To the best of our knowledge, little previous work in the literature models semantics and structure of threaded discussions simultaneously. However, there are still some previous works that should be investigated, as they are related to mining threaded discussions. Most of them focus on knowledge acquisition from web forums. We can again categorize their methods into semantic models and structure models.

For the semantic models, [7, 6] targeted at extracting and ranking answers for given questions in web forums based on their content information. Methods in [12], [10] and [19] mine the relationship among posts, relationship between post and thread title, and relationship between post and a thread based on their content similarity, respectively.

For the structure models, FGrank [8] automatically generates topic hierarchy, constructs page level link graph based on topic hierarchy, modifies the PageRank algorithm in order to rank forum pages. But it cannot represent the characteristic of reply relationship among posts. EABIF [20] is another work related to structure models. It built the influence network among consumers based on the time order of consumption records. It then proposed how to model the influence diffusion. We believe that it is also possible to apply this model in threaded discussions. The above studies implement the semantic decomposition and structure reconstruction in two phases, which conflict with our assumptions that structure in discussion threads usually changes along with semantics. Consequently, the reconstructed structure by prior research is not consistent with the evolving nature of semantics in threaded discussions.

3. A SIMULTANEOUS MODEL OF SEMANTICS AND STRUCTURE

In this section, we present the details of SMSS. First, we introduce some useful concepts in threaded discussions.

DEFINITION 1 (POST). *A post is a user-submitted content at a particular time stamp. Posts are contained in threads, where they appear as boxes one after another. If a post is not the first of the corresponding thread, it is referred to as a reply.*

DEFINITION 2 (THREAD). *A discussion thread (without ambiguity we use thread hereafter) is a series of posts, usually displayed—by default—from the oldest to the latest. A*

thread is initiated by a root post which may contain news, questions, opinions, and so on; and is followed by a number of non-root posts. A post other than the root one must reply to one of its previous posts in the same thread.

Intuitively, a threaded discussion has the following four characteristics. We take the thread in Fig. 1 as an example.

A discussion thread has several topics. The example in Fig. 1 mainly discusses two topics, namely “Silverlight 1.0” and “Javascript”. Accordingly, we hope we can find a semantic representation of each post, such representation demonstrates a mixture of topics in the post. Suppose there are T topics and V words; the j^{th} topic can be described as a distribution in the word space \mathbb{R}^V , as $\vec{x}^{(j)} \in \mathbb{R}^V, 1 \leq j \leq T$. The i^{th} post $\vec{d}^{(i)}$ is assumed to be a mixture of various topics, as:

$$\vec{d}^{(i)} \simeq \sum_{j=1}^T \vec{\theta}_j^{(i)} \cdot \vec{x}^{(j)}$$

where $\vec{\theta}_j^{(i)}$ is the coefficient of post $\vec{d}^{(i)}$ on topic $\vec{x}^{(j)}$. To estimate the topic space $X = \{\vec{x}^{(1)}, \dots, \vec{x}^{(T)}\}$, we can minimize the loss function:

$$\|D - X\Theta\|_F^2$$

Here the thread contains L posts as $D = \{\vec{d}^{(1)}, \dots, \vec{d}^{(L)}\}$; and the coefficient matrix $\Theta = \{\vec{\theta}^{(1)} \dots \vec{\theta}^{(L)}\}$.

An individual post is related to a few topics. Although one thread may contain several semantic topics, each individual post usually concentrates on a limited number of topics. For example, the 2nd post and the 3rd post in Fig. 1 are only related to the “Silverlight 1.0” topic. Thus we may assume that the coefficient vector for each post is very sparse and introduce an L1 sparse regularizer:

$$\|\vec{\theta}^{(i)}\|_1$$

A post is related to its previous posts. When a user joins a thread, he/she usually reads those existing posts in the thread. If he/she is interested in some previous posts, he/she may write down their comments. Thus the semantics of the reply post is related to its previous posts, which reflects the structural characteristics of a thread. For example in Fig. 1, the 7th post is related to both the 5th and 6th posts. We can formally represent such structural constraint as a regularizer:

$$\|\vec{\theta}^{(i)} - \sum_{k=1}^{i-1} \vec{b}_k^{(i)} \cdot \vec{\theta}^{(k)}\|_F^2$$

where $\vec{b}_k^{(i)}$ is the structural coefficient between the i^{th} post and the k^{th} post and which shows how the k^{th} post affects the i^{th} post. The fact that $\vec{\theta}^{(i)}$ corresponds to the post $\vec{d}^{(i)}$ can be approximated by a linear combination of $\vec{\theta}^{(k)}$ in previous posts.

The reply relations are sparse. Note that in real scenarios, users usually intend to comment on one or two previous posts, for example in Fig. 1, although there are a lot of posts before the 6th post, it is only related to the 5th post. We again introduce an L1 regularizer to favor sparse structural coefficients:

$$\|\vec{b}^{(i)}\|_1$$

Based on the above description, we define the SMSS model as follows. Given the post matrix D , topic number T , the goal of thread modeling is to estimate the value of the topic matrix X , the coefficient matrix Θ , and the structural coefficients b of each post, by minimizing the following loss function f :

$$\begin{aligned}
f = & \|D - X\Theta\|_F^2 + \lambda_1 \sum_{i=1}^L \|\bar{\theta}^{(i)}\|_1 \\
& + \lambda_2 \sum_{i=1}^L \|\bar{\theta}^{(i)} - \sum_{k=1}^{i-1} b_k^{(i)} \cdot \bar{\theta}^{(k)}\|_F^2 + \lambda_3 \sum_{i=1}^L \|\bar{b}^{(i)}\|_1 \quad (1)
\end{aligned}$$

In (1), the first term encourages the post to be reconstructed well from topics; the second term constraints the topic representation to be sparse for each post; the third term encourages the post to be approximated from previous posts within the same thread; and the fourth term constrains that the structural coefficients should be sparse too. The optimization objective balances the four terms by parameters λ_1 , λ_2 , and λ_3 . In this way, both the semantic and the structural information are estimated simultaneously.

Suppose we have a collection of M threads which shared the same topic matrix X , we can optimize them together by minimizing the loss functions over all the threads to obtain a globally optimal topic representation and structural coefficients as follows:

$$\text{minimize}_{X, \{\Theta^{(n)}\}, \{\bar{b}^{(i)(n)}\}} \sum_{n=1}^M f^{(n)}(\cdot) \quad (2)$$

Although the optimization problem is not jointly convex, we can still solve it by iteratively minimizing the convex sub-problems. For more details about the optimization, please refer to the appendix section.

4. APPLICATIONS

Towards the three problems in threaded discussions, from Section 4.1 to Section 4.3, we conduct three experiments to answer the following questions:

- Is our model capable of recognizing structural information among posts? To answer this, we show the result of reply relationship reconstruction.
- Is our model capable of capturing semantics? To answer this, we show the result of junk identification.
- Is our model capable of measuring content quality? To answer this, we show the result of expert finding, since the core problem of finding experts is to estimate the overall quality of user-generated content.

Actually, the proposed solutions are straightforwardly implemented based on the proposed **SMSS** model.

4.1 Reply Reconstruction

We reconstruct reply relationships among posts based on both their semantics and structures estimated by the **SMSS** model. Intuitively, posts with reply relations should have similar terms. Thus the task of finding reply relations can be converted into a text retrieval task. However, the major challenge here is that the length of each post is usually very short and there are usually few common words between two posts. Topic similarity is more robust and interpretable by reducing high-dimensional term representation to lower-dimensional latent topics. However, only topic similarity itself may lose some detailed information. Combining topic similarity with term similarity is more efficient for short and sparse text and has been verified in [17]. Our idea here is to integrate term similarity, topic similarity, and structural similarity. The structural similarity can be seen as a lower-dimensional representations of topics. Since each individual post may only focus on one or two topics, there are still few common topics shared by two posts. Here, the structural similarity acts as a smoother of topic similarity; and

Algorithm 1 Reply-Reconstruction($D^{(n)}$, ρ). Given the n^{th} thread with length L , $D^{(n)}$ is the set of posts and $D^{(n)} = \{\bar{d}^{1(n)}, \dots, \bar{d}^{L(n)}\}$. ρ is the threshold. $Rep^{(n)}$ is the reply structure.

```

1: Order  $D^{(n)}$  based on their posted times
2: for  $i = 1$  to  $L$  do
3:   for  $j = 1$  to  $i$  do
4:     compute  $sim(\bar{d}^{i(n)}, \bar{d}^{j(n)})$ 
5:   end for
6:    $c = \arg \max_j sim(\bar{d}^{i(n)}, \bar{d}^{j(n)})$ 
7:   if  $sim(\bar{d}^{i(n)}, \bar{d}^{c(n)}) \geq \rho$  then
8:      $Rep_i^{(n)} = \bar{d}^{c(n)}$ 
9:   else
10:     $Rep_i^{(n)} = \bar{d}^{i(n)}$ 
11:   end if
12: end for
13: Return the reply structure  $Rep^{(n)}$ 

```

we will show its improvement in the experiments. Formally, the similarity of a post j and an early post i is defined as:

$$\begin{aligned}
sim(i, j) = & sim(\bar{d}^{(i)}, \bar{d}^{(j)}) + w_1 \cdot sim(\bar{\theta}^{(i)}, \bar{\theta}^{(j)}) \\
& + w_2 \cdot sim(\bar{b}^{(i)}, \bar{b}^{(j)}) \quad (3)
\end{aligned}$$

where the similarity in each component is the cosine value of two corresponding feature vectors. We use w_1 and w_2 as the ratios to balance the three components.

Based on this similarity measure, we propose an algorithm to analyze a thread with L posts, as shown in Algorithm 1. That is, for a new unlabeled post we compute the similarity between itself and each of its previous posts, rank the similarities, and then choose the top ranked post as a candidate parent. In case that the candidate parent is not similar enough to the new post, we assume that the new post initializes a new branch of the thread.

4.2 Junk Identification

A primary issue in online discussion threads is junk. For example, the fourth post in Fig. 1 is a typical chitchat post. Moreover, junk has become a focus of community administrators, users, and even developers, because junk content increases the cost of maintaining a clean and healthy communication environment, and distracts users from readings [15].

A discussion thread usually focuses on a very limited number of topics, while junk posts usually have different topics and act as outliers. Moreover, junk content is similar and common among various threads. To detect just posts, we introduce a background topic in the model. To construct the background topic, we select some common words by their thread frequencies in the whole data corpus as:

$$\bar{x}_w^{(bg)} = |\{V_w : V_w \in \bar{d}^i\}|/|D| \quad (4)$$

where $|D|$ is the total number of posts in the corpus and $|\{V_w : V_w \in \bar{d}^i\}|$ represents the number of posts where the term V_w appears. In (2), the topic matrix X is shared by all threads. We define and normalize the background topic $\bar{x}^{(bg)}$ as the last column of X . When optimizing (2), we fix $\bar{x}^{(bg)}$ in the optimization process.

Finally, we propose a straightforward criterion of junk detection based on the topic coefficients $\bar{\theta}^{(i)}$ of each post. The probability of a post i being a junk is defined as:

$$p_{junk}(i) = \bar{\theta}_{bg}^{(i)} / \sum_t \bar{\theta}_t^{(i)} \quad (5)$$

In this way, posts close to the background topic are very likely to be junk. It is worth noting that in the SMSS model, the topic projection is also affected by structural constraints, and provides more accurate description of each post. We will show this advantage in the experiments by comparing some other models which only characterize a post according to its semantics.

4.3 Expert Finding

Online communities have become important places for people to seek and share expertise. The reply relationships among posts actually construct a reply network. Studies on such a reply network tell us that reply structure in a discussion thread can help to evaluate users’ expertise [22]. That is, posts written by experts are usually of high quality and thus attract more users’ attention and more replies. However, experiments in [22] were conducted on a particular site which recorded the reply relationships among posts in each discussion thread; while for most community sites, such as those forums powered by vBulletin and phpBB, there are few explicit reply relations. Fortunately, using the SMSS model and the Algorithm 1, we can approximately recover the implicit reply structure in a thread, and consequently construct a reply network G as:

$$G = (N, E), E = \{e_{i,j}, \forall i, j, \text{user } n_i \text{ replies user } n_j\} \quad (6)$$

where N is a set of nodes of G , each node corresponds to a user in the forum. E is the set of directed edges of G . The weight of the edge is the number of posts replied to this user.

Inspired by [22], we employ the HITS [13] algorithm as a straightforward method to rank the users. Through this task, we try to demonstrate the capability of the reconstructed reply relations from SMSS model. The experimental results show that the performance of our method is even better than that based on the original reply relations. We will explain it detailedly in experiment part. It suggests that this method can be extended to more web communities without explicit reply structures. We will give more detailed explanation in the next section.

5. EXPERIMENTS

In this section, we present experimental results of our model in three applications. We implemented all algorithms in C# and all experiments have been executed on a server with an AMD Opteron Processor 280 2.40 GHz (4 cores) and main memory 8G bytes.

5.1 Data Set

We use two forums, Apple Discussion (discussions.apple.com) and Slashdot (www.slashdot.org), as our data sources. Apple discussion center is designed for Apple fans to post questions of Apple products and discussions of Apple new trends. Slashdot is a forum for developers, game fans, and all kinds of users to freely comment on recent technical news and events. These two forums are carefully selected because of the following reasons:

- These two forums provide explicit reply relations, which can be used as the ground truth to evaluate the performance of reply reconstruction.
- These two forums have reliable judgements for post qualities. In Slashdot, the moderators mark each post according to its quality. The score ranges from $-1 \sim 5$, where higher score refers to informative and insightful

Table 1: Descriptive statistics of data sets

Data	Slashdot	Apple
Number of threads	1154	4486
Number of posts	203210	80008
Average thread length	176.09	17.84
Average words per post	73.53	78.36
Number of topics	5	5
Average posts per user	15.32	4.69

posts. In Apple, users post questions, and mark helpful and correct replies in which their questions are solved.

- These two forums provide clear topic categories. In Apple, each thread belongs to one board of a specific topic. In Slashdot, most threads are tagged by users with topic phrases.

We selected the largest 5 topics from each forums and filtered out unqualified threads, such as short threads which contain only one post, un-rated threads, and non-tagged threads in Slashdot. The statistic results are shown in Table 1, from which we can conclude that the two forums differ with respect to several attributes. Apple threads are shorter “Question & Answer” style discussions, with average 17.84 posts, Slashdot threads are longer chatting-style communications, with an average length of 176.09 posts. Slashdot users are more active with 15.32 posts per user while in Apple there are about 4.69 posts per user. However, posts in threaded discussions are very terse with about 70 words on average. The following experiments are done without stemming and filtering stopwords.

5.2 Reply Reconstruction

First, we evaluate the performance of reply reconstruction. We manually wrote a wrapper to parse pages and extract exact reply relations as the ground truth. The threshold used in Algorithm 1 was tuned based on a small data set, as $\rho = 0.4$, and it performed stably well for both Apple Discussion and Slashdot. Although there are some reply hints in the titles in Slashdot, to show the generalization ability of our method, we did not use this kind of information in reply reconstruction. The evaluation metric is precision. We did not measure recall since we assigned the reply relations for all posts.

For comparison, we adopted some naive methods such as Nearest-Previous (NP), Reply-Root (RR), and Only Document Similarity (DS). NP assigns each post to its nearest previous post as the reply target; RR assigns each post to the root post as the reply target; and DS assigns each post to the post which has most similar terms. Moreover, we also compared our SMSS model with some state-of-the-art models which can provide semantic topic analysis, such as latent Dirichlet allocation (LDA) [3] and the special words with background model (SWB) [5]. We computed the post similarity by $sim(i, j) = sim(\vec{d}^{(i)}, \vec{d}^{(j)}) + w_1 \cdot sim(\vec{\theta}_{LDA}^{(i)}, \vec{\theta}_{LDA}^{(j)})$ from LDA. SWB is an extension of LDA, by allowing words in documents to be modeled either from general topics, or from post-specific “special” word distributions, or from a thread-wide background distribution. We leverage the topic distributions $\vec{\theta}_{SWB}^{(i)}$ and special-words distributions $\vec{\psi}_{SWB}^{(i)}$ for similarity computing, as $sim(i, j) = sim(\vec{d}^{(i)}, \vec{d}^{(j)}) + w_1 \cdot sim(\vec{\theta}_{SWB}^{(i)}, \vec{\theta}_{SWB}^{(j)}) + w_2 \cdot sim(\vec{\psi}_{SWB}^{(i)}, \vec{\psi}_{SWB}^{(j)})$.

In the experiments, the combination weights w_i for SMSS, LDA, and SWB were tuned based on a small set of data (about 60 threads from Slashdot and Apple, respectively). We only plot the performance trends of different parameters

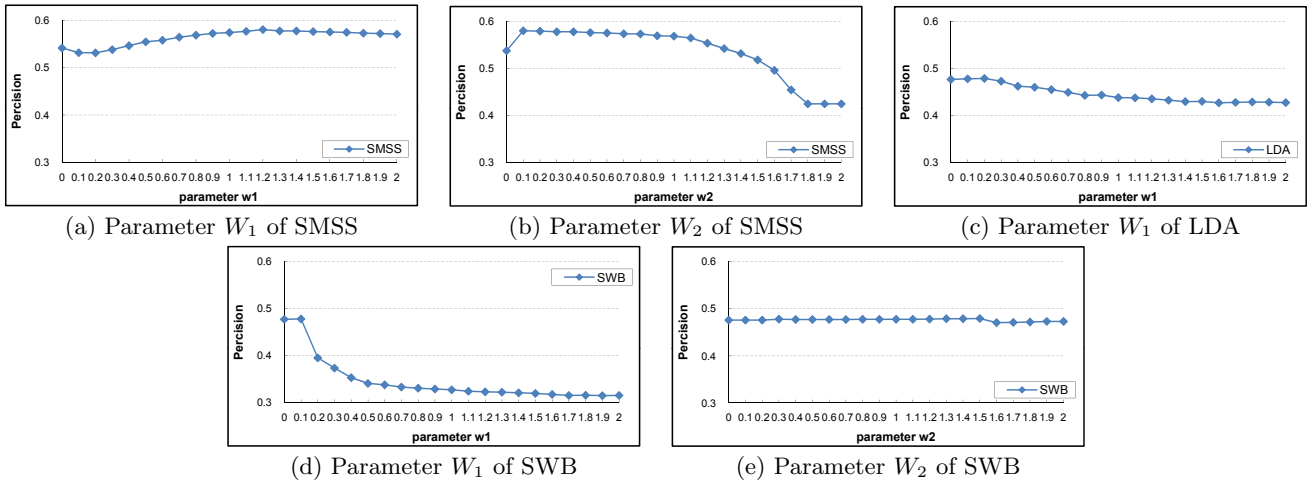


Figure 2: Performance of reply reconstruction when using different parameters of SMSS, LDA, and SWB.

Table 2: Performance of reply reconstruction in all posts v.s. high-quality posts

Method	Slashdot		Apple	
	All Posts	Good Posts	All posts	Good Posts
NP	0.021	0.012	0.289	0.239
RR	0.183	0.319	0.269	0.474
DS	0.463	0.643	0.409	0.628
LDA	0.465	0.644	0.410	0.648
SWB	0.463	0.644	0.410	0.641
SMSS	0.524	0.737	0.517	0.772

for Slashdot in Figure 2 due to space limitation. The evaluation was conducted based on the rest part of the data. The results are shown in Table 2.

From Figure 2 and Table 2, we have four observations: (I) in Slashdot, a certain number of posts reply to their thread root, few posts reply to their nearest previous posts; while in Apple Discussion, there are almost equal number of posts replying to the nearest previous neighbors and the thread roots. This is caused by different discussion styles of the two forums. Discussion threads in Apple Discussion follow a Question-Answer style. New solutions and fresh questions in replies invoke a series of discussions. However, threads in Slashdot are usually initialized by a piece of news. Interesting aspects of the news and brilliant replies arise branches of discussions. (II) The best parameters of topic similarity of SWB and LDA are both $w_1 = 0.1$ and the improvements are very small comparing with baseline *DS*. The best parameter of topic similarity of SMSS is $w_1 = 1.2$. Thus the topics of SMSS are more capable of characterizing reply relations. (III) In our experiments, SWB achieves the best performance when w_2 is very small and different w_2 s have little effect to the performance. This is because posts in threaded discussions are usually short and it is very difficult to estimate a sound coefficient for document specific word distribution. (IV) SMSS demonstrates great improvement. A major difference between SMSS and other approaches is that SMSS reconstructs the structure representation $b^{(i)}$ for post p_i in each discussion thread. Though the best parameter of structural similarity is $w_2 = 0.1$, the improvement is remarkable (from 0.538 to 0.580). This indicates that, besides semantic similarities, structure similarities are more distinguishing in identifying reply relations.

Furthermore, we analyze the performance on posts with different qualities. We chose good posts whose scores are

larger than 3 in Slashdot and posts marked as ‘‘Helpful’’ or ‘‘Solved’’ in Apple Discussion (about 10% among all posts). The similarity-based methods have better performance for those posts with high quality. It does make sense since posts with high quality may cause more significant replies. We will show its benefit in the following expert finding application.

5.3 Junk Identification

In this subsection, we evaluate the performance of junk identification. We only used the threads from Slashdot in this task because useless posts were not explicitly annotated in Apple Discussion. To make the results more comparable among different methods, we only selected threads with more than 60 posts in the experiment. We set the posts whose score are -1 or 0 as junk posts in the ground truth. We use precision, recall, and F-Measure to measure the performance.

An intuitive idea to identify junk post is based on the statistical information of common words, as: $\eta(p_i) = \sum_w df(w)/|p_i|$, where $|p_i|$ is the length of document, w is a word in p_i , $df(w)$ is the document frequency of word w in the corpus. One post is marked as a junk post if $\eta(p_i)$ is larger than the average η in the corpus. We call this intuitive method DF and use it as the baseline. The SWB model also integrates background into topics, so we use it as another comparative method. To detect junk posts, we follow (5) where the $\bar{\theta}_{bg}^{(i)}$ is produced by SWB. Moreover, as junk identification can be regarded as a binary classification task, we also compared our model with SVM. For SVM training, We selected 2000 posts (no overlap with test data) as training set, in which posts with score 0 and -1 are positive samples, and posts with score 4 and 5 are negative samples. Features for SVM are terms with tf-idf weighting.

As shown in Table 3, SMSS outperforms the other comparative methods. The reasons are listed below: (I) Junk posts are thread-dependent. There are few common junk words across different threads. Hence SVM’s performance is not good. (II) SMSS is built on the statistical information of DF. But SMSS puts structural constraints during the process of projecting posts into topic space. Hence SMSS outperforms DF in terms of precision, recall and F-Measure. (III) SWB learns background that fits the corpus and thus achieves high precision. However, it does not have structural constraints. When a post can not be reconstructed well from previous posts, SMSS assigns smaller coefficients to topics,

Table 3: Performance of junk identification

Method	Prec.	Recall	F-Measure
SWB	0.48	0.22	0.30
SVM	0.37	0.24	0.20
DF	0.34	0.40	0.36
SMSS	0.38	0.45	0.41

resulting in a larger ratio of p_{junk} . Thus SMSS can detect more outlier posts, and obtain higher recall and F-Measure.

5.4 Expert Finding

In the subsection, we evaluate the performance of expert search. Since there is no explicit user-supplied expertise ranking data in the two forums, we need to generate a “gold standard” as the ground truth for evaluation. It is impractical to manually rate a large number of these users, in the experiments we followed the sophisticated standard from the Internet Movie Database (IMDB)³. In this standard, the formula for calculating each user’s rating score gives a true Bayesian estimate:

$$Rating = \frac{n}{n+m} \cdot S + \frac{m}{n+m} \cdot AvgS \quad (7)$$

where n is the number of posts of a user and S is the average score of the user’s posts; m is the threshold of the minimum posts for a user to be an expert candidate, and $AvgS$ is the mean score of the posts from all the users. The advantage of this estimation formula is that it balances the bias and uncertainty of users with less posts. For users with less posts, the rating is more likely to be pushed toward the mean score $AvgS$; for users with more posts, the rating is more likely to be the true average score of the user’s own posts. In the experiment, we obtained the top 100 experts according to the rating. Moreover, the expert ranking was carried out independently for each boards/topics, since some experts may only focus on a small number of fields. Accordingly, the m and $AvgS$ in (7) are selected for each topic respectively. The experiments were done in Apple Discussion and Slashdot respectively. We calculated the average performance for each method.

To analyze the reply network and find out experts, we employed three typical structural models: HITS [13], PageRank [16], and EABIF [20]. HITS and PageRank have been widely adopted in measuring node quality in networks; while EABIF adopts an adjusted mechanism of PageRank in the influence network. EABIF assumes that experts diffuse information to others, and long path of propagation will cause information lost. The decreasing factor is $\beta^{N-1}/(N-1)!$ where N is the length of propagation path and $\beta = 2$. To demonstrate the effectiveness of the re-constructed reply structures from SMSS, we implemented all structural models on both the reply network in ground truth (“Original” in Table 4) and the network re-constructed by SMSS (“Reconstructed” in Table 4).

Moreover, we choose one state-of-the-art language model (LM) in [1] (we use the model 2, document model, since it outperforms model 1 in nearly all situations in [1]) for comparison. Here, each post is treated as a document without structure. LM estimates the probability of one candidate ca being an expert in the topic t in a collection of posts.

$$p(ca|t) = \sum_{d \in P_{ca}} \prod_{q \in t} \{(1 - \lambda_d)p(q|d) + \lambda_d p(q)\}^{n(q,t)} \times p(d|ca)$$

³<http://www.imdb.com/chart/top>

Table 4: Performance of expert finding

Method	MRR.	MAP	P@10
LM	0.821	0.698	0.800
EABIF (Original)	0.674	0.362	0.243
EABIF (Reconstructed)	0.742	0.318	0.281
PageRank (Original)	0.675	0.377	0.263
PageRank (Reconstructed)	0.743	0.321	0.266
HITS (Original)	0.906	0.832	0.900
HITS (Reconstructed)	0.938	0.822	0.906

where P_{ca} is the post written by ca , q is a word in the topic, the smooth parameter $\lambda_d = \beta/(\beta+n(d))$, λ_d is proportional to the post length $n(d)$, and β is the average post length in the data set.

Given the expert ground truth (7), the evaluation metrics, including Mean Reciprocal Rank (MRR), Mean Average Precision (MAP), and Precision at top 10 results (P@10), are calculated for each model/method on each topic; and the average performances are shown in Table 4. We have the following observations: (I) LM gives a relatively high performance but it is still not good enough. This is because in threaded discussions, some posts are highly relevant to the given topic but doesn’t contain insightful information, e.g. posts describing naive questions. LM cannot distinguish such posts since it only measures expertise by the number of relevant posts. (II) The performance of HITS is significantly better than PageRank and EABIF. In our opinion, it is because of the different content qualities of posts and their replies. In HITS, these two scores are treated separately (hub/authority scores) while PageRank and EABIF treat them equally. (III) Structural models perform consistently better on reconstructed reply network than the original network in terms of MRR and P@10; and perform worse on reconstructed reply network than on original network in terms of MAP. MRR and P@10 measure precision on the top results while MAP measures average precision. The reply reconstruction algorithm has higher performance in identifying replies to high quality posts (as shown in Section 5.2). Algorithm 1 filters out noisy data, prune faint replies. Thus those users receiving more significant replies will be pushed toward a higher position, which leads to a better performance (and higher confidence) in the top results. For a real experts finding task, we argue that the top results are more important, because people usually do not have the patience to browse through the whole returned list.

6. CONCLUSIONS

Threaded discussions are valuable data sources with lots of human knowledge in various domains. In this paper, we have presented a sparse coding-based model, to simultaneously representing semantics and structure of threaded discussions. By adding sparse constraints that each post is generated from only a few topics and each reply is related to only a few previous posts, the proposed SMSS model has advantages in three perspectives: (I) it can characterize mutual information between semantics and structure in discussion threads by modeling them simultaneously; (II) it can help identify junk posts more accurately to avoid their disturbance in content analysis; and (III) it can help find experts in a given board/sub-board (topic). We demonstrated the competency of SMSS with these three applications. The results show promising performance in various situations.

Although the results are encouraging, there is still room for further improvements. The method of identifying junk posts is straightforward, we will try other approaches in the

future. We will also try to explain this model in a probabilistic framework.

7. ACKNOWLEDGMENTS

The first author is partially supported by Shanghai Leading Academic Discipline Project, Project number: B114.

8. REFERENCES

- [1] K. Balog, L. Azzopardi, and M. de Rijke. A language modeling framework for expert finding. *Information Processing and Management*, 06(003):1–12, 2008.
- [2] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *Proc. of ICML*, pages 113–120, 2006.
- [3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(6):993–1022, 2003.
- [4] W. Buntine and A. Jakulin. Applying discrete PCA in data analysis. In *Proc. of UAI*, pages 59–66, 2004.
- [5] C. Chemudugunta, P. Smyth, and M. Steyvers. Modeling general and specific aspects of documents with a probabilistic topic model. *Advances in neural information processing systems*, 41(6):391–407, 1990.
- [6] G. Cong, L. Wang, C.-Y. Lin, Y.-I. Song, and Y. Sun. Finding question-answer pairs from online forums. In *Proc. 31st SIGIR*, pages 467–474, 2008.
- [7] S. Ding, G. Cong, C.-Y. Lin, and X. Zhu. Using conditional random fields to extract contexts and answers of questions from online forums. In *Proc. 11th ACL*, pages 710–718, 2008.
- [8] X. Gu and W.-Y. Ma. Building implicit links from content for forum search. In *Proc. 29th SIGIR*, pages 300–307, 2006.
- [9] T. Hofmann. Probabilistic latent semantic indexing. In *Proc. 29th SIGIR*, pages 50–57, 1999.
- [10] J. Huang, M. Zhou, and D. Yang. Extracting chatbot knowledge from online discussion forums. In *Proc. 11th IJCAI*, pages 423–428, 2006.
- [11] J. Scott. *Social Network Analysis: A Handbook*. Sage Publications, London, 2000.
- [12] J. W. Kim, K. S. Candan, and M. E. Donderler. Topic segmentation of message hierarchies for indexing and navigation support. In *Proc. 16th WWW*, pages 322–331, 2005.
- [13] J. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–622, 1999.
- [14] A. McCallum, A. Corrada-Emmanuel, and X. Wang. Topic and role discovery in social networks. In *Proc. of IJCAI*, pages 249–272, 2007.
- [15] G. Mishne, D. Carmel, and R. Lempel. Blocking blog spam with language model disagreement. In *Proc. of AIRWeb*, 2005.
- [16] L. Page, S. Brin, R. Motwani, and T. Winograd. *The PageRank citation ranking: Bringing order to the web. Technical report*. Stanford University, 1998.
- [17] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proc. of WWW*, pages 91–100, 2008.
- [18] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *Proc. of UAI*, pages 487–494, 2004.
- [19] D. Shen, Q. Yang, J.-T. Sun, and Z. Chen. Thread detection in dynamic text message streams. In *Proc. 29th SIGIR*, pages 35–42, 2006.

- [20] X. Song, B. L. Tseng, C.-Y. Lin, and M.-T. Sun. Personalized recommendation driven by information flow. In *Proc. 29th SIGIR*, pages 509–516, 2006.
- [21] C. Wang, D. M. Blei, and D. Heckerman. Continuous time dynamic topic models. In *Proc. of UAI*, 2008.
- [22] J. Zhang, M. S. Ackerman, and L. Adamic. Expertise networks in online communities: structure and algorithms. In *Proc. of WWW*, pages 221–230, 2007.

APPENDIX

In this section, we give an approximate solution to the sparse coding model. First we write the objective function in equation 1 in the matrix form. In the scope of appendix, we use $\vec{\theta}^{(i)}$ to denote the i^{th} column vector in Θ , $\vec{x}^{(i)}$ to denote the i^{th} row vector in X , $\vec{b}^{(i)}$ to denote the corresponding column vector for the i^{th} post, and $b_j^{(i)}$ to denote the coefficient of previous post j to current post i . Then we rewrite (1) as:

$$f = \|D - X\Theta\|_F^2 + \lambda_1 \sum_i \|\vec{\theta}^{(i)}\|_1 + \lambda_2 \sum_i \|\vec{\theta}^{(i)} - \theta^{(<i)} \vec{b}^{(i)}\|_2^2 + \lambda_3 \sum_i \|\vec{b}^{(i)}\|_1$$

where $\theta^{(<i)}$ is a $T \times i$ matrix indicating the first i columns in Θ .

When Θ and X are fixed, for each $\vec{b}^{(i)}$:

$$\frac{\partial f}{\partial \vec{b}^{(i)}} = -\lambda_2 \theta^{(<i)T} (\vec{\theta}^{(i)} - \theta^{(<i)} \vec{b}^{(i)}) + \lambda_3 \text{sign}(\vec{b}^{(i)})$$

Let $\frac{\partial f}{\partial \vec{b}^{(i)}} = 0$, we have:

$$\vec{b}^{(i)} = \frac{\text{sign}(\theta^{(<i)T} \vec{\theta}^{(i)}) (\lambda_2 |\theta^{(<i)T} \vec{\theta}^{(i)}| - \lambda_3)}{\lambda_2 \theta^{(<i)T} \theta^{(<i)}} \quad (8)$$

When b and X are fixed, for each $\vec{\theta}^{(i)}$:

$$\frac{\partial f}{\partial \vec{\theta}^{(i)}} = -\vec{x}^{(i)T} (D - X\Theta) + \lambda_1 \text{sign}(\vec{\theta}^{(i)}) + \lambda_2 (\vec{\theta}^{(i)} - \theta^{(<i)} \vec{b}^{(i)}) - \sum_{j=i+1} b_i^{(j)} (\vec{\theta}^{(j)} - \theta^{(<j)} \vec{b}^{(j)})$$

Let $x^{(-i)} \theta^{(-i)} = \sum_{k=1, k \neq i} \vec{x}^{(k)} \vec{\theta}^{(k)}$, we have:

$$X\Theta = \vec{x}^{(i)} \vec{\theta}^{(i)} + x^{(-i)} \theta^{(-i)}$$

Let $\frac{\partial f}{\partial \vec{\theta}^{(i)}} = 0$, and

$$G = \vec{x}^{(i)T} D - \vec{x}^{(i)T} x^{(-i)} \theta^{(-i)} + \lambda_2 \theta^{(<i)} \vec{b}^{(i)} + \lambda_2 \sum_{j=i+1} b_i^{(j)} (\vec{\theta}^{(j)} - b_{-i}^{(j)} \theta_{-i}^{(<j)})$$

We have:

$$\vec{\theta}^{(i)} = \frac{\text{sign}(G) (|G| - \lambda_1)}{\vec{x}^{(i)T} \vec{x}^{(i)} + \lambda_2 + \lambda_2 \sum_{j=i+1} (b_i^{(j)})^2} \quad (9)$$

When Θ and all \vec{b} are fixed, suppose D is the term matrix of all M threads, we can optimize X by:

$$X = D^M \Theta^{-1} \quad (10)$$

At the beginning, we assign a random initial value to X and normalize the matrix X , and then we repeat the optimization loop for a fix round c . In each round, we first optimize b in all threads by (8), Θ in all threads by (9), and X by (10) in sequence and then normalize X .