Topic Sentiment Mixture: Modeling Facets and Opinions in Weblogs

Qiaozhu Mei[†], Xu Ling[†], Matthew Wondra[†], Hang Su[‡], ChengXiang Zhai[†]
[†]Department of Computer Science
University of Illinois at Urbana-Champaign
[‡]Department of EECS
Vanderbilt University

ABSTRACT

In this paper, we define the problem of topic-sentiment analysis on Weblogs and propose a novel probabilistic model to capture the mixture of topics and sentiments simultaneously. The proposed Topic-Sentiment Mixture (TSM) model can reveal the latent topical facets in a Weblog collection, the subtopics in the results of an ad hoc query, and their associated sentiments. It could also provide general sentiment models that are applicable to any ad hoc topics. With a specifically designed HMM structure, the sentiment models and topic models estimated with TSM can be utilized to extract topic life cycles and sentiment dynamics. Empirical experiments on different Weblog datasets show that this approach is effective for modeling the topic facets and sentiments and extracting their dynamics from Weblog collections. The TSM model is quite general; it can be applied to any text collections with a mixture of topics and sentiments, thus has many potential applications, such as search result summarization, opinion tracking, and user behavior prediction.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Text Mining

General Terms: Algorithms

Keywords: topic-sentiment mixture, weblogs, mixture model, topic models, sentiment analysis

1. INTRODUCTION

More and more internet users now publish online dairies and express their opinions with Weblogs (i.e., blogs). The wide coverage of topics, dynamics of discussion, and abundance of opinions in Weblogs make blog data extremely valuable for mining user opinions about all kinds of topics (e.g., products, political figures, etc.), which in turn would enable a wide range of applications, such as opinion search for ordinary users, opinion tracking for business intelligence, and user behavior prediction for targeted advertising.

Technically, the task of mining user opinions from Weblogs boils down to sentiment analysis of blog data – identifying and extracting positive and negative opinions from blog articles. Although much work has been done recently on blog mining [11, 7, 6, 15], most existing work aims at extracting and analyzing topical contents of blog articles without any analysis of sentiments in an article. The lack of sentiment analysis in such work often limits the effectiveness of the mining results. For example, in [6], a burst of blog mentions about a book has been shown to be correlated with a spike of sales of the book in Amazon.com. However, a burst of *criticism* of a book is unlikely to indicate a growth of the book sales. Similarly, a decrease of blog mentions about a product might actually be caused by the decrease of *complaints* about its defects. Thus understanding the positive and negative opinions about each topic/subtopic of the product is critical to making more accurate predictions and decisions.

There has also been some work trying to capture the positive and negative sentiments in Weblogs. For example, Opinmind [20] is a commercial weblog search engine which can categorize the search results into positive and negative opinions. Mishne and others analyze the sentiments [18] and moods [19] in Weblogs, and use the temporal patterns of sentiments to predict the book sales as opposed to simple blog mentions. However, a common deficiency of all this work is that the proposed approaches extract only the overall sentiment of a query or a blog article, but can neither distinguish different subtopics within a blog article, nor analyze the sentiment of a subtopic. Since a blog article often covers a mixture of subtopics and may hold different opinions for different subtopics, it would be more useful to analyze sentiments at the level of subtopics. For example, a user may like the *price* and *fuel efficiency* of a new Toyota Camry, but dislike its *power* and *safety* aspects. Indeed, people tend to have different opinions about different features of a product [28, 13]. As another example, a voter may agree with some points made by a presidential candidate, but disagree with some others. In reality, a general statement of good or bad about a query is not so informative to the user, who usually wants to drill down in different facets and explore more detailed information (e.g., "price", "battery life", "warranty" of a laptop). In all these scenarios, a more in-depth analysis of sentiments in specific aspects of a topic would be much more useful than the analysis of the overall sentiment of a blog article.

To improve the accuracy and utility of opinion mining from blog data, we propose to conduct an in-depth analysis of blog articles to reveal the major topics in an article, associate each topic with sentiment polarities, and model the dynamics of each topic and its corresponding sentiments. Such topic-sentiment analysis can potentially support many applications. For example, it can be used to generate a more detailed topic-sentiment summary of Weblog search results as shown in Figure 1.

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

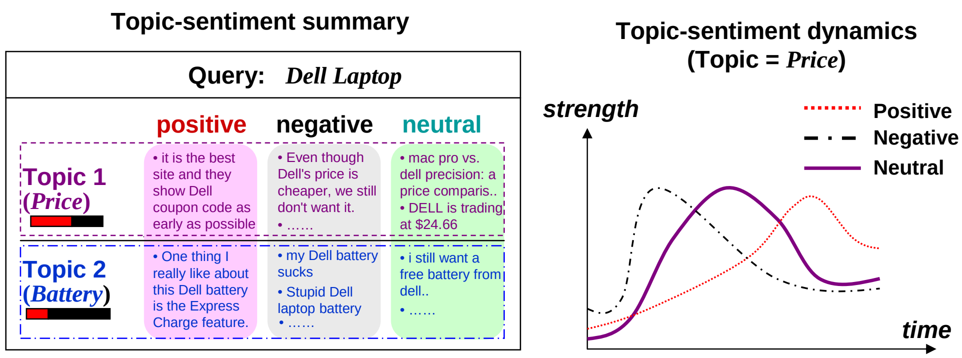


Figure 1: A possible application of topic-sentiment analysis

In Figure 1, given a query word representing a user's ad hoc information need (e.g., a product), the system extracts the latent facets (subtopics) in the search results, and associates each subtopic with positive and negative sentiments. From the example sentences on the left, which are organized in a two dimensional structure, the user can understand the pros and cons of each facet of the product, or what are its best and worst aspects. From the strength dynamics of a topic and its associated sentiments on the right, the user can get deeper understanding of how the opinions about a specific facet change over time. To the best of our knowledge, no existing work could simultaneously extract multiple topics and different sentiments from Weblog articles.

In this paper, we study the novel problem of modeling subtopics and sentiments simultaneously in Weblogs. We formally define the **Topic-Sentiment Analysis** (TSA) problem and propose a probabilistic mixture model called **Topic-Sentiment Mixture** (TSM) to model and extract the multiple subtopics and sentiments in a collection of blog articles. Specifically, a blog article is assumed to be "generated" by sampling words from a mixture model involving a background language model, a set of topic language models, and two (*positive* and *negative*) sentiment language models. With this model, we can extract the topic/subtopics from blog articles, reveal the correlation of these topics and different sentiments, and further model the dynamics of each topic and its associated sentiments. We evaluate our approach on different weblog data sets. The results show that our method is effective for all the tasks of the topic-sentiment analysis.

The proposed approach is quite general and has many potential applications. The mining results are quite useful for summarizing search results, monitoring public opinions, predicting user behaviors, and making business decisions. Our method requires no prior knowledge about a domain, and can extract general sentiment models applicable to any ad hoc queries. Although we only tested the TSM on Weblog articles, it is applicable to any text data with mixed topics and sentiments, such as customer reviews and emails.

The rest of the paper is organized as follows. In Section 2, we formally define the problem of Topic-Sentiment Analysis. In Section 3, we present the Topic-Sentiment Mixture model and discuss the estimation of its parameters. We show how to extract the dynamics of topics and sentiments in Section 4, and present our experiment results in Section 5. In Sections 6 and 7, we discuss the related work and conclude.

2. PROBLEM FORMULATION

In this section, we formally define the general problem of Topic-Sentiment Analysis.

Let $\mathcal{C} = \{d_1, d_2, ..., d_m\}$ be a set of documents (e.g., blog articles). We assume that $\mathcal{C}$ covers a number of topics, or subtopics (also known as *themes*) and some related sentiments. Following [9, 1, 16, 17], we further assume that there are k major topics (subtopics) in the documents, $\{\theta_1, \theta_2, ..., \theta_k\}$, each being characterized by a multinomial distribution over all the words in our vocabulary (also known as a unigram language model). Following [23, 21, 13], we assume that there are two sentiment polarities in Weblog articles, the *positive* and the *negative* sentiment. The two sentiments are associated with each topic in a document, representing the positive and negative opinions about the topic.

Definition 1 (Topic Model) A ***topic model*** θ in a text collection $\mathcal{C}$ is a probabilistic distribution of words $\{p(w|\theta)\}_{w\in V}$ and represents a semantically coherent topic. Clearly, we have $\sum_{w\in V} p(w|\theta) = 1$.

Intuitively, the high probability words of a topic model often suggest what theme the topic captures. For example, a topic about the movie "Da Vinci Code" may assign a high probability to words like "movie", "Tom" and "Hanks" This definition can be easily extended to a distribution of multiword phrases. We assume that there are k such topic models in the collection.

Definition 2 (Sentiment Model) A ***sentiment model*** in a text collection $\mathcal{C}$ is a probabilistic distribution of words representing either *positive* opinions ($\{p(w|\theta_P)\}_{w\in V}$) or *negative* opinions ($\{p(w|\theta_N)\}_{w\in V}$). We have $\sum_{w\in V} p(w|\theta_P) = 1$ and $\sum_{w\in V} p(w|\theta_N) = 1$.

Sentiment models are orthogonal to topic models in the sense that they would assign high probabilities to general words that are frequently used to express sentiment polarities whereas topical models would assign high probabilities to words representing topical contents with neutral opinions.

Definition 3 (Sentiment Coverage) A ***sentiment coverage*** of a topic in a document (or a collection of documents) is the relative coverage of the neurtral, positive, and negative opinions about the topic in the document (or the collection of documents). Formally, we define a sentiment coverage of topic θ_i in document d as $c_{i,d} = \{\delta_{i,d,F}, \delta_{i,d,P}, \delta_{i,d,N}\}$. $\delta_{i,d,F}$, $\delta_{i,d,P}$, $\delta_{i,d,N}$ are the coverage of neutral, positive, and negative opinions, respectively; they form a probability distribution and satisfy $\delta_{i,d,F} + \delta_{i,d,P} + \delta_{i,d,N} = 1$.

In many applications, we also want to know how the neutral discussions, the positive opinions, and the negative opinions about the topic (subtopic) change over time. For this purpose, we introduce two additional concepts, "topic life cycle" and "sentiment dynamics" as follows.

Definition 4 (Topic Life Cycle) A ***topic life cycle***, also known as a *theme life cycle* in [16], is a time series representing the strength distribution of the neutral contents of a topic over the time line. The strength can be measured based on either the amount of text which a topic can explain [16] or the relative strength of topics in a time period [15, 17]. In this paper, we follow [16] and model the topic life cycles with the amount of document content that is generated with each topic model in different time periods.

Definition 5 (Sentiment Dynamics) The ***sentiment dynamics*** for a topic θ is a time series representing the strength distribution of a sentiment $s \in \{P, N\}$ associated with θ. The strength can indicate how much positive/negative opinion there is about the given topic in each time period. Being consistent with topic life cycles, we model the sentiment dynamics with the amount of text associated with topic θ that is generated with each sentiment model.

Based on the concepts above, we define the major tasks of **Topic-Sentiment Analysis** (**TSA**) on weblogs as: **(1) Learning General Sentiment Models:** Learn a sentiment model for positive opinions and a sentiment model for negative opinions, which are general enough to be used in new unlabeled collections. **(2) Extracting Topic Models and Sentiment Coverages:** Given a collection of Weblog articles and the general sentiment models learnt, customize the sentiment models to this collection, extract the topic models, and extract the sentiment coverages. **(3) Modeling Topic Life Cycle and Sentiment Dynamics:** Model the life cycles of each topic and the dynamics of each sentiment associated with that topic in the given collection.

This problem as defined above is more challenging than many existing topic extraction tasks and sentiment classification tasks for several reasons. First, it is not immediately clear how to model topics and sentiments simultaneously with a mixture model. No existing topic extraction work [9, 1, 16, 15, 17] could extract sentiment models from text, while no sentiment classification algorithm could model a mixture of topics simultaneously. Second, it is unclear how to obtain sentiment models that are independent of specific contents of topics and can be generally applicable to any collection representing a user's ad hoc information need. Most existing sentiment classification methods overfit to the specific training data provided. Finally, computing and distinguishing topic life cycles and sentiment dynamics is also a challenging task. In the next section, we will present a unified probabilistic approach to solve these challenges.

3. A MIXTURE MODEL FOR THEME AND SENTIMENT ANALYSIS

3.1 The Generation Process

A lot of previous work has shown the effectiveness of mixture of multinomial distributions (mixture language models) in extracting topics (themes, subtopics) from either plain text collections or contextualized collections [9, 1, 16, 15, 17, 12]. However, none of this work models topics and sentiments simultaneously; if we apply an existing topic model on the weblog articles directly, none of the topics extracted with this model could capture the positive or negative sentiment well.

To model both topics and sentiments, we also use a mixture of multinomials, but extend the model structure to include two sentiment models to naturally capture sentiments.

In the previous work [15, 17], the words in a blog article are classified into two categories: (1) common English words (e.g., "the", "a", "of") and (2) words related to a topical theme (e.g., "nano", "price", "mini" in the documents about iPod). The common English words are captured with a background component model [28, 16, 15], and the topical words are captured with topic models. In our topic-sentiment model, we extend the categories for the topical words in existing approaches. Specifically, for the words related to a topic, we further categorize them into three sub-categories: (1) words about the topic with neutral opinions (e.g., "nano", "price"); (2) words representing the positive opinions of the topic (e.g., "awesome", "love"); and (3) words representing the negative opinions about the topic (e.g., "hate", "bad"). Correspondingly, we introduce four multinomial distributions: (1) θ_B is a background topic model to capture common English words; (2) $\Theta = \{\theta_1, ..., \theta_k\}$ are k topic models to capture neutral descriptions about k global subtopics in the collection; (3) θ_P is a positive sentiment model to capture positive opinions; and (4) θ_N is a negative sentiment model to capture negative opinions for all the topics in the collection.

According to this mixture model, an author would "write" a Weblog article by making the following decisions stochastically and sampling each word from the component models: (1) The author would first decide whether the word will be a common English word. If so, the word would be sampled according to θ_B. (2) If not, the author would then decide which of the k subtopics the word should be used to describe. (3) Once the author decides which topic the word is about, the author will further decide whether the word is used to describe the topic neutrally, positively, or negatively. (4) Let the topic picked in step (2) be the j-th topic θ_j. The author would finally sample a word using θ_j, θ_P or θ_N, according to the decision in step(3). This generation process is illustrated in Figure 2.

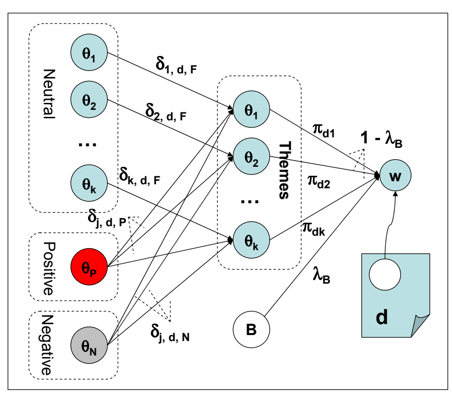


Figure 2: The generation process of the topic-sentiment mixture model

We now formally present the Topic-Sentiment Mixture model and the estimation of parameters based on blog data.

3.2 The Topic-Sentiment Mixture Model

Let $\mathcal{C} = \{d_1, ..., d_m\}$ be a collection of weblog articles, $\Theta = \{\theta_1, ..., \theta_k\}$ be k topic models, θ_P and θ_N be a positive and negative sentiment model respectively. The log likelihood of the whole collection $\mathcal{C}$ according to the TSM model is

$$\begin{aligned} log(\mathcal{C}) &= \sum_{d \in \mathcal{C}} \sum_{w \in V} c(w:d) log[\lambda_B p(w|B) + (1-\lambda_B) \sum_{j=1}^{k} \pi_{dj} \times \\ & (\delta_{j,d,F} p(w|\theta_j) + \delta_{j,d,P} p(w|\theta_P) + \delta_{j,d,N} p(w|\theta_N))] \end{aligned}$$

where $c(w:d)$ is the count of word w in document d, λ_B is the probability of choosing θ_B, π_{dj} is the probability of choosing the j-th topic in document d, and $\{\delta_{j,d,F}, \delta_{j,d,P}, \delta_{j,d,N}\}$ is the sentiment coverage of topic j in document d, as defined in Section 2.

Similar to existing work [28, 16, 15, 17], we also regularize this model by fixing some parameters. λ_B is set to an empirical constant between 0 and 1, which indicates how much noise that we believe exists in the weblog collection. We then set the background model as

$$p(w|\theta_B) = \frac{\sum_{d \in \mathcal{C}} c(w,d)}{\sum_{w \in V} \sum_{d \in \mathcal{C}} c(w,d)}$$

The parameters remaining to be estimated are: (1) the topic models, $\Theta = \{\theta_1, ..., \theta_k\}$; (2) the sentiment models, θ_P and θ_N; (3) the document topic probabilities π_{dj}; and (4) the sentiment coverage for each document, $\{\delta_{j,d,F}, \delta_{j,d,P}, \delta_{j,d,N}\}$. We denote the whole set of free parameters as Λ.

Without any prior knowledge, we may use the maximum likelihood estimator to estimate all the parameters. Specifically, we can use the Expectation-Maximization (EM) algorithm [3] to compute the maximum likelihood estimate iteratively; the updating formulas are shown in Figure 3. In these formulas, $\{z_{d,w,j,s}\}$ is a set of hidden variables ($s \in \{F, P, N\}$), and $p(z_{d,w,j,s})$ is the probability that word w in document d is generated from the j-th topic, using topic/sentiment model s.

However, in reality, if we do not provide any constraint on the model, the sentiment models estimated from the EM algorithm will be very biased towards specific contents of the collection, and the topic models will also be "contaminated" with sentiments. This is because the opinion words and topical words may co-occur with each other, thus they will not be separated by the EM algorithm. This is unsatisfactory as we want our sentiment models to be independent of the topics, while the topic models should be neutral. In order to solve this problem, we introduce a regularized two-phase estimation framework, in which we first learn a general prior distribution on the sentiment models and then combine this prior with the data likelihood to estimate the parameters using the maximum a posterior (MAP) estimator.

3.3 Defining Model Priors

The prior distribution should tell the TSM what the sentiment models should look like in the working collection. This knowledge may be obtained from domain specific lexicons, or training data in this domain as in [23]. However, it is impossible to have such knowledge or training data for every ad hoc topics, or queries. Therefore, we want the prior sentiment models to be general enough to apply to any ad hoc topics. In this section, we show how we may exploit an online sentiment retrieval service such as Opinmind [20] to induce a general prior on the sentiment models. When given a query, Opinmind can retrieve positive sentences and negative sentences, thus we can obtain examples with sentiment labels for a topic (i.e., the query) from Opinmind. The query can be regarded as a topic label. To ensure diversity of topics, we can submit various queries to Opinmind and mix all the results to form a training collection. Presumably, if the topics in this training collection are diversified enough, the sentiment models learnt would be very general.

With such a training collection, we have topic labels and sentiment labels for each document. Formally, we have $\mathcal{C} = \{(d, t_d, s_d)\}$, where t_d indicates which topics the document is about, and s_d indicates whether d holds positive or negative opinions about the topics. We then use the topic-sentiment model presented in Section 3.2 to fit the training data and estimate the sentiment models. Since we have topic and sentiment labels, we impose the following constraints: (1) $\pi_{dj} = 1$ if $t_d = j$ and $\pi_{dj} = 0$ otherwise; (2) $\delta_{j,d,P} = 0$ if s_d is *negative* and $\delta_{j,d,N} = 0$ if s_d is *positive*.

In Section 5, we will show that this estimation method is effective for extracting general sentiment models and the diversity of topics helps improve the generality of the sentiment models learnt.

Rather than directly using the learnt sentiment models to analyze our target collection, we use them to define a prior on the sentiment models and estimate sentiment models (and the topic models) using the maximum a posterior estimator. This way would allow us to adapt the general sentiment models to our collection and further improve the accuracy of the sentiment models, which is traditionally done in a domain dependent way. Specifically, let $\bar{\theta}_P$ and $\bar{\theta}_N$ be the positive and negative sentiment models learnt from some training collections. We define the following two conjugate Dirichlet priors for the sentiment model θ_P and θ_N, respectively: $Dir(\{1 + \mu_P p(w|\bar{\theta}_P)\}_{w \in V})$ and $Dir(\{1 + \mu_N p(w|\bar{\theta}_N)\}_{w \in V})$, where the parameters μ_P and μ_N indicate how strong our confidence is on the sentiment model prior. Since the prior is conjugate, μ_P (or μ_N) can be interpreted as "equivalent sample size", which means that the impact of adding the prior would be equivalent to adding $\mu_P p(w|\bar{\theta}_P)$ (or $\mu_N p(w|\bar{\theta}_N)$) pseudo counts for word w when estimating the sentiment model $p(w|\theta_P)$ (or $p(w|\theta_N)$).

If we have some prior knowledge on the topic models, we can also define them as conjugate prior for some θ_j. Indeed, given a topic, a user often has some knowledge about what aspects are interesting. For example, when the user is searching for laptops, we know that he is very likely interested in "price" and "configuration". It will be nice if we "guide" the model to enforce two of the topic models to be as close as possible to the predefined facets.

Therefore, in general, we may assume that the prior on all the parameters in the model is

$$\begin{aligned} p(\Lambda) &\propto p(\theta_P) * p(\theta_N) * \prod_{j=1}^{k} p(\theta_j) = \prod_{w \in V} p(w|\theta_P)^{\mu_P p(w|\bar{\theta}_P)} \\ & \prod_{w \in V} p(w|\theta_N)^{\mu_N p(w|\bar{\theta}_N)} \prod_{j=1}^{k} \prod_{w \in V} p(w|\theta_j)^{\mu_j p(w|\bar{\theta}_j)} \end{aligned}$$

where $\mu_j = 0$ if we do not have prior knowledge on θ_j.

3.4 Maximum A Posterior Estimation

With the prior defined above, we may use the MAP estimator: $\hat{\Lambda} = \arg\max_{\Lambda} p(\mathcal{C}|\Lambda) p(\Lambda)$

$$p(z_{d,w,j,F}=1) = \frac{(1-\lambda_B)\pi_{dj}^{(n)}\delta_{j,d,F}^{(n)}p^{(n)}(w|\theta_j)}{\lambda_B p(w|\theta_B)+(1-\lambda_B)\sum_{j'=1}^{k}\pi_{dj}^{(n)}(\delta_{j',d,F}^{(n)}p^{(n)}(w|\theta'_j)+\delta_{j',d,P}^{(n)}p^{(n)}(w|\theta_P)+\delta_{j',d,N}^{(n)}p^{(n)}(w|\theta_N))}$$

$$p(z_{d,w,j,P}=1) = \frac{(1-\lambda_B)\pi_{dj}^{(n)}\delta_{j,d,P}^{(n)}p^{(n)}(w|\theta_P)}{\lambda_B p(w|\theta_B)+(1-\lambda_B)\sum_{j'=1}^{k}\pi_{dj}^{(n)}(\delta_{j',d,F}^{(n)}p^{(n)}(w|\theta'_j)+\delta_{j',d,P}^{(n)}p^{(n)}(w|\theta_P)+\delta_{j',d,N}^{(n)}p^{(n)}(w|\theta_N))}$$

$$p(z_{d,w,j,N}=1) = \frac{(1-\lambda_B)\pi_{dj}^{(n)}\delta_{j,d,N}^{(n)}p^{(n)}(w|\theta_N)}{\lambda_B p(w|\theta_B)+(1-\lambda_B)\sum_{j'=1}^{k}\pi_{dj}^{(n)}(\delta_{j',d,F}^{(n)}p^{(n)}(w|\theta'_j)+\delta_{j',d,P}^{(n)}p^{(n)}(w|\theta_P)+\delta_{j',d,N}^{(n)}p^{(n)}(w|\theta_N))}$$

$$\pi_{dj}^{(n+1)} = \frac{\sum_{w\in V}c(w,d)(p(z_{d,w,j,F}=1)+p(z_{d,w,j,P}=1)+p(z_{d,w,j,N}=1))}{\sum_{j'=1}^{k}\sum_{w\in V}c(w,d)(p(z_{d,w,j',F}=1)+p(z_{d,w,j',P}=1)+p(z_{d,w,j',N}=1))}$$

$$\delta_{j,d,F}^{(n+1)} = \frac{\sum_{w\in V}c(w,d)p(z_{d,w,j,F}=1)}{\sum_{w\in V}c(w,d)(p(z_{d,w,j,F}=1)+p(z_{d,w,j,P}=1)+p(z_{d,w,j,N}=1))}$$

$$\delta_{j,d,P}^{(n+1)} = \frac{\sum_{w\in V}c(w,d)p(z_{d,w,j,P}=1)}{\sum_{w\in V}c(w,d)(p(z_{d,w,j,F}=1)+p(z_{d,w,j,P}=1)+p(z_{d,w,j,N}=1))}$$

$$\delta_{j,d,N}^{(n+1)} = \frac{\sum_{w\in V}c(w,d)p(z_{d,w,j,N}=1)}{\sum_{w\in V}c(w,d)(p(z_{d,w,j,F}=1)+p(z_{d,w,j,P}=1)+p(z_{d,w,j,N}=1))}$$

$$p^{(n+1)}(w|\theta_j) = \frac{\sum_{d\in\mathcal{C}}c(w,d)p(z_{d,w,j,F}=1)}{\sum_{w'\in V}\sum_{d\in\mathcal{C}}c(w',d)p(z_{d,w',j,F}=1)}$$

$$p^{(n+1)}(w|\theta_P) = \frac{\sum_{d\in\mathcal{C}}\sum_{j=1}^{k}c(w,d)p(z_{d,w,j,P}=1)}{\sum_{w'\in V}\sum_{d\in\mathcal{C}}\sum_{j=1}^{k}c(w',d)p(z_{d,w',j,P}=1)}$$

$$p^{(n+1)}(w|\theta_N) = \frac{\sum_{d\in\mathcal{C}}\sum_{j=1}^{k}c(w,d)p(z_{d,w,j,N}=1)}{\sum_{w'\in V}\sum_{d\in\mathcal{C}}\sum_{j=1}^{k}c(w',d)p(z_{d,w',j,N}=1)}$$

Figure 3: EM updating formulas for the topic-sentiment mixture model

It can be computed by rewriting the M-step in the EM algorithm in Section 3.2 to incorporate the pseudo counts given by the prior [14]. The new M-step updating formulas are:

$$p^{(n+1)}(w|\theta_P) = \frac{\mu_P p(w|\bar{\theta}_P)+\sum_{d\in\mathcal{C}}\sum_{j=1}^{k}c(w,d)p(z_{d,w,j,P}=1)}{\mu_P+\sum_{w'\in V}\sum_{d\in\mathcal{C}}\sum_{j=1}^{k}c(w',d)p(z_{d,w',j,P}=1)}$$

$$p^{(n+1)}(w|\theta_N) = \frac{\mu_N p(w|\bar{\theta}_N)+\sum_{d\in\mathcal{C}}\sum_{j=1}^{k}c(w,d)p(z_{d,w,j,N}=1)}{\mu_N+\sum_{w'\in V}\sum_{d\in\mathcal{C}}\sum_{j=1}^{k}c(w',d)p(z_{d,w',j,N}=1)}$$

$$p^{(n+1)}(w|\theta_j) = \frac{\mu_j p(w|\bar{\theta}_j)+\sum_{d\in\mathcal{C}}c(w,d)p(z_{d,w,j,F}=1)}{\mu_j+\sum_{w'\in V}\sum_{d\in\mathcal{C}}c(w',d)p(z_{d,w',j,F}=1)}$$

The parameters $\mu's$ can be either empirically set to constants, or set through regularized estimation [25], in which we would start with very large $\mu's$ and then gradually discount $\mu's$ in each EM iteration until some stopping condition is satisfied.

3.5 Utilizing the Model

Once the parameters in the model are estimated, many tasks can be done by utilizing the model parameters.

1. Rank sentences for topics: Given a set of sentences and a theme j, we can rank the sentences according to a topic j with the score

$$Score_j(s) = -D(\theta_j||\theta_s) = -\sum_{w\in V}p(w|\theta_j)\log\frac{p(w|\theta_j)}{p(w|\theta_s)}$$

where θ_s is a smoothed language model of sentence s.

2. Categorize sentences by sentiments: Given a sentence s assigned to topic j, we can assign s to positive, negative, or neutral sentiment according to

$$\arg\max_x -D(\theta_s||\theta_x) = \arg\max_x -\sum_{w\in V}p(w|\theta_s)\log\frac{p(w|\theta_s)}{p(w|\theta_x)}$$

where $x \in \{j, P, N\}$, and θ_s is a language model of s.

3. Reveal the overall opinions for documents/topics: Given a document d and a topic j, the overall sentiment distribution for j in d is the sentiment coverage $\{\delta_{j,d,F}, \delta_{j,d,P}, \delta_{j,d,N}\}$. The overall sentiment strength (e.g., positive sentiment) for the topic j is

$$S(j,P) = \frac{\sum_{d\in\mathcal{C}}\pi_{dj}\delta_{j,d,P}}{\sum_{d\in\mathcal{C}}\pi_{dj}}$$

4. SENTIMENT DYNAMICS ANALYSIS

While the TSM model can be directly used to analyze topics and sentiments in many ways, it does not directly model the topic life cycles or sentiment dynamics. In addition to associating the sentiments with multiple subtopics, we would also like to show how the positive/negative opinions about a given subtopic change over time. The comparison of such temporal patterns (i.e., topic life cycles and corresponding sentiment dynamics) could potentially provide more in-depth understanding of the public opinions than [20], and yield more accurate predictions of user behavior than using the methods proposed in [6] and [19].

To achieve this goal, we can approximate these temporal patterns by partitioning documents into their corresponding time periods and computing the posterior probability of $p(t|\theta_j)$, $p(t|\theta_j, \theta_P)$ and $p(t|\theta_j, \theta_N)$, where t is a time period. This approach has the limitation that these posterior distributions are not well defined, because the time variable t is nowhere involved in the original model. An alternative approach would be to model the time variable t explicitly in the model as in [15, 17], but this would bring in many more free parameters to the model, making it harder to estimate all the parameters reliably. Defining a good partition of the time line is also a challenging problem, since too coarse a partition would miss many bursting patterns, while too fine granularity a time period may not be estimated reliably because of data sparseness.

In this work, we present another approach to extract topic life cycles and sentiment dynamics, which is similar to the method used in [16]. Specifically, we use a hidden Markov model (HMM) to tag every word in the collection with a topic and sentiment polarity. Once all words are tagged, the topic life cycles and sentiment dynamics could be extracted by counting the words with corresponding labels.

We first sort the documents with their time stamps, and convert the whole collection into a long sequence of words. On the surface, it appears that we could follow [16] and construct an HMM with each state corresponding to a topic model (including the background model), and set the output probability of state j to $p(w|\theta_j)$. A topic state can either stay on itself or transit to some other topic states through the background state. The system can learn (from our collection) the transition probabilities with the Baum-Welch algorithm [24] and decode the collection sequence with the Viterbi algorithm [24]. We can easily model sentiments by adding two sentiment states to the HMM. Unfortunately, this structure cannot decode which sentiment word is about which topic. Below, we present an alternative HMM structure (shown in Figure 4) that can better serve our purpose.

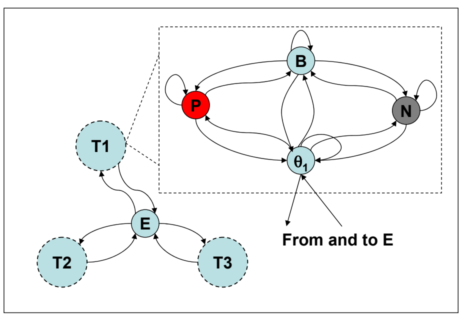


Figure 4: The Hidden Markov Model to extract topic life cycles and sentiment dynamics

In Figure 4, state E controls the transitions between topics. In addition to E, there are a series of pseudo states, each of which corresponds to a subtopic. These pseudo states can only transit from each other through state E. A pseudo state is not a real single state, but a substructure of states and transitions. For example, pseudo state T1 consists of four real states, three of them correspond to the topic model θ_1, the positive sentiment model, and the negative sentiment model. The remaining state corresponds to the background model. The four states in each pseudo state are fully connected, except that there is no direct transition between two sentiment states. The output probabilities of all states (except for state E) are fixed according to the corresponding topic or sentiment models. This HMM structure can decode both topic segments (with pseudo state T_j) and sentiment segments associated with each topic (with states inside T_j).

We force the model to start with state E, and use the Baum-Welch algorithm to learn the transition probabilities and the output probability for E. Once all the parameters are estimated, we use the Viterbi algorithm to decode the collection sequence. Finally, as in [16], we compute the topic life cycles and sentiment dynamics by counting the number of words labeled with the corresponding state over time.

5. EXPERIMENTS AND RESULTS

5.1 Data Sets

We need two types of data sets for evaluation. One is used to learn the general sentiment priors, thus should have labels for positive and negative sentiments. In order to extract very general sentiment models, we want the topics in this data set to be as diversified as possible. We construct this training data set by leveraging an existing weblog sentiment retrieval system (i.e., Opinmind.com [20]), i.e., we submit different queries to Opinmind and mix the downloaded classified results. This also gives us natural boundaries of topics in the training collection. The composition of this training data set (denotated as "OPIN") is shown in Table 1.

Topic	# Pos.	# Neg.	Topic	# Pos.	# Neg.
laptops	346	142	people	441	475
movies	396	398	banks	292	229
universities	464	414	insurances	354	297
airlines	283	400	nba teams	262	191
cities	500	500	cars	399	334

Table 1: Basic statistics of the OPIN data sets

The other type of data is used to evaluate the extraction of topic models, topic life cycles, and sentiment dynamics. Such data do not need to have sentiment labels, but should have time stamps, and be able to represent users' ad hoc information needs. Following [16], we construct these data sets by submitting time-bounded queries to Google Blog Search [1] and collect the blog entries returned. We restrict the search domain to spaces.live.com, since schema matching is not our focus. The basic information of these test collections (notated as "TEST") is shown in Table 2.

Data Set	# doc.	Time Period	Query Term
iPod	2988	1/11/05∼11/01/06	ipod
Da Vinci Code	1000	1/26/05∼10/31/06	da+vinci+code

Table 2: Basic statistics of the TEST data sets

For all the weblog collections, Krovetz stemmer [10] is used to stem the text.

5.2 Sentiment Model Extraction

Our first experiment is to evaluate the effectiveness of learning the prior models for sentiments. As discussed in Section 3.3, a good $\bar{\theta}_s$ should not be dependent with the specific features of topics, and be general enough to be used to guild the learning of sentiment models for unseen topics. The more diversified the topics of the training set are, the more general the sentiment models estimated should be.

To evaluate the effectiveness of our TSM model on this task, we collect labeled results of 10 different topics from Opinmind, each of which consists of an average of 5 queries. We then construct a series of training data sets, such that for any k ($1 \leq k \leq 9$), there are 10 training data sets, each of which is a mixture of k topics. We then apply the TSM model on each data set and extract sentiment models

[1]http://blogsearch.google.com

accordingly. We also construct a dataset with the mixture of all 10 topics. The top words of the sentiment models which are extracted from the 10-topic-mixture data set and those from a single-topic data set are compared in Table 3.

P-mix	**N-mix**	P-movies	N-movies	P-cities	N-cities
love	suck	love	hate	beautiful	hate
awesome	hate	harry	harry	love	suck
good	stupid	pot	pot	awesome	people
miss	ass	brokeback	mountain	amaze	traffic
amaze	fuck	mountain	brokeback	live	drive
pretty	horrible	awesome	suck	good	fuck
job	shitty	book	evil	night	stink
god	crappy	beautiful	movie	nice	move
yeah	terrible	good	gay	time	weather
bless	people	watch	bore	air	city
excellent	evil	series	fear	greatest	transport

Table 3: Sentiment models learnt from a mixture of topics are more general

The left two columns in Table 3 present the two sentiment models extracted from the 10-topic-mixture dataset, which is more general than the right two columns and two in the middle, which are extracted from two single-topic data sets ("movies" and "cities") respectively. In the two columns in the middles, we see terms like "harry", "pot", "brokeback", "mountain" ranked highly in the sentiment models. These words are actually part of our query terms. We also see other domain specific terms such as "movie", "series", "gay", and "watch". In the sentiment models from "cities" dataset, we remove all query terms from the top words. However, we could still notice words like "night", "air" in the positive model, and "traffic", "weather", "transport" in the negative model. This indicates that the sentiment models are highly biased towards the specific features of the topic, if the training data set only contains one topic.

To evaluate this in a more principled way, we conduct a 10-fold cross validation, which numerically measures the closeness of the sentiment models learnt from a mixture of topics ($k = 1 \sim 9$), and those from an unseen topic (i.e., a topic not in the mixture). Intuitively, a sentiment model is less biased if it is closer to unseen topics. The closeness of two sentiment models is measured with the Kullback-Leibler(KL) Divergence, the formula of which is

$$D(\theta_x||\theta_y) = \sum_{w \in V} p(w|\theta_x) \log \frac{p(w|\theta_x)}{p(w|\theta_y)}$$

where θ_x and θ_y are two sentiment models (e.g., θ_P learnt from a mixture-topic collection and θ_P from a single-topic collection). We use a simple Laplace smoothing method to guarantee $p(w|\theta_y) > 0$. The result of the cross validation is presented in Figure 5.

Figure 5 measures the average KL divergence between the positive (negative) sentiment model learnt from a k-topic-mixture dataset and the positive (negative) sentiment model learnt from an unseen single-topic dataset. We notice that when k is larger, where the topics in the training dataset are more diversified, the sentiment models learnt from the collection are closer to the sentiment models of unseen topics. This validates our assumption that a more diversified training collection could provide more general sentiment prior models for new topics. The sentiment models ($\bar{\theta}_P$ and $\bar{\theta}_N$) estimated from the 10-topic-mixture collection are used as the prior sentiment models in the following experiments.

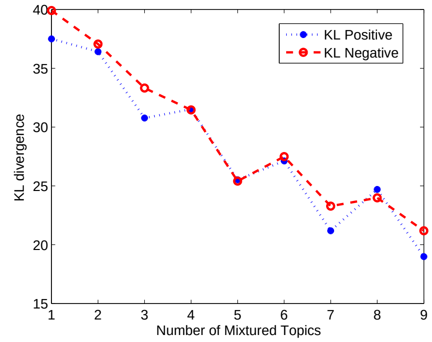


Figure 5: Sentiment model leant from diversified topics better fits unseen topics

5.3 Topic Model Extraction

Our second experiment is to fit the TSM to ad hoc weblog collections and extract the topic models and sentiment coverage. As discussed in Section 3.3, the general sentiment models learnt from the OPIN Data set will be used as a strong prior for the sentiment models in a given collection. We expect the topic models extracted be unbiased towards sentiment polarities, which simply represent the neutral contents of the topics.

In the experiments, we set the initial values of $\mu's$ reasonably large (>10,000), and use the regularized estimation strategy in [25] to gradually decay the $\mu's$. λ_B is empirically set between $0.85 \sim 0.95$. Some informative topic models extracted from the TEST data sets are shown in Table 4, 5.

NO-Prior			**With-Prior**	
batt., nano	**marketing**	**ads, spam**	**Nano**	**Battery**
battery	apple	free	nano	battery
shuffle	microsoft	sign	color	shuffle
charge	market	offer	thin	charge
nano	zune	freepay	hold	usb
dock	device	complete	model	hour
itune	company	virus	4gb	mini
usb	consumer	freeipod	dock	life
hour	sale	trial	inch	rechargeable

Table 4: Example topic models with TSM: iPod

NO-Prior			**With-Prior**	
content	**book**	**background**	**movie**	**religion**
langdon	author	jesus	movie	religion
secret	idea	mary	hank	belief
murder	holy	gospel	tom	cardinal
louvre	court	magdalene	film	fashion
thrill	brown	testament	watch	conflict
clue	blood	gnostic	howard	metaphor
neveu	copyright	constantine	ron	complaint
curator	publish	bible	actor	communism

Table 5: Example topic models: Da Vinci Code

As discussed in Section 3.4, we can either extract topic models in a completely unsupervised way, or base on some prior of what the topic models should look like. In Table 4, 5, the left three columns are topic models extracted without prior knowledge, and the right columns are those extracted with the bold titles as priors. We see that the topics extracted in either way are informative and coherent. The ones extracted with priors are extremely clear and distinctive, such as "Nano" and "battery" for the query "iPod".

This is quite desirable in summarizing search results, where the system could extract topics in an interactive way with the user. For example, the user can input several words as expected facets, and the system uses these words (e.g., "movie", "book", "history" for the query "Da Vinci Code" and "battery", "price" for the query "iPod") as prior on some topic models, and let the remaining to be extracted in the unsupervised way.

With the topic models and sentiment models extracted, we can summarize the sentences in blog search results, by first ranking sentences according to different topics, and then assigning them into sentiment categories. Table 6 shows the summarized results for the query "Da Vinci Code".

We show two facets of the results: "movie" and "book". Although both the movie and the book are named as "The Da Vinci Code", many people hold different opinions about them. Table 6 well organizes sentences retrieved for the query "da vinci code" by the relevance to each facets, and the categorization as positive, negative, and neutral opinions. The sentences do not have to contain the facet name, such as "Tom Hanks stars in the movie". The bolded sentence clearly presents an example of mixed topics. We also notice that the system sometimes make the wrong classifications. For example, the sentence "anybody is interested in it?" is misclassified as positive. This is because we rely on a unigram language model for the sentiments, and the "bag of words" assumption does not consider word dependency and linguistics. This problem can be tackled when phrases are used as the bases of the sentiment models.

In Table 7, we compare the query summarization of our model to that of Opinmind. The left two columns are summarized search results with TSM and right two columns are top results from Opinmind, with the same query "iPod". We see that Opinmind tends to rank sentences with strongest sentiments to the top, but many of which are not very informative. For example, although the sentences "I love iPod" and "I hate iPod" do reflect strong attitudes, they do not give the user as useful information as "out of battery again". Our system, on the other hand, reveals the hidden facets of people's opinions. In the results from Opinmind, we do notice that some sentences are about specific aspects of iPod, such as "battery", "video", "microsoft (indicating marketing)". Unfortunately, these useful information are mixed together. Our system organizes the sentences according to the hidden aspects, which provides the user a deeper understanding of the opinions about the query.

5.4 Topic Life Cycle and Sentiment Dynamics

Based on the topic models and sentiment models learnt from the TEST collections, we evaluate the effectiveness of the HMM based method presented in Section 4, on extraction of topic life cycles and sentiment dynamics.

Intuitively, we expect the sentiment models to explain as much information as possible, since the most useful patterns are sentiment dynamics. In our experiments, we force the transition probability from topic states to sentiment states, and those from sentiment models to themselves to be reasonably large (e.g., >0.25). The results of topic life cycles and sentiment dynamics are selectively presented in Figure 6.

In Figure 6(a) and (b), we present the dynamics of two facets in the Da-Vinci-Code Collection: "book" and "religion, belief". The neutral line in each plot corresponds to the topic life cycle. In both plots, we see a significant burst in May, 2006, which was caused by the release of the movie "Da Vinci Code". However, before the movie, we can still notice some bursts of discussions about the book. In the plot of the "book" facet, the positive sentiments consistently dominates the opinions during the burst. For the religion issues and the conflicts of the belief, however, the negative opinions are stronger than the positive opinions during the burst, which is consistent with the heated debates about the movie around that period of time. In fact, the book and movie are boycotted or even banned in some countries because of the confliction to their religious belief.

In Figure 6(c) and (d), we present the topic life cycle and the sentiment dynamics of the subtopic "Nano" for "iPod". In Figure 6(c), we see that both the neutral topic and the positive sentiment about Nano burst around early September, 2005. That is consistent with the time of the official introduction of iPod Nano. The negative sentiment, however, does not burst until several weeks later. This is reasonable, since people need to experience the product for a while before discovering its defects. In Figure 6(d), we alternatively plot the relative strength of positive (negative) sentiment over the negative (positive) sentiment. This relative strength clearly reveals which sentiment dominates the opinions, and the trend of this domination. Since there are generally less negative opinions, we plot the Neg/Pos line with a different scale. Again, we see that around the time that the iPod Nano was introduced, the positive sentiments dominate the opinions. However, in October 2005, the negative sentiments shows a sudden increase of coverage. This overlaps with the time period in which there was a bursting of complaints, followed by a lawsuit about the "scratch problem" of iPod Nano.

We also plot the Pos/Neg dynamics of the overall sentiments about "iPod". We see that its shape is much different than the Pos/Neg plot of "Nano". The positive sentiment holds a larger proportion of opinions, but this domination is getting weaker. This also suggests that it is not reasonable to use the overall blog mentions (not distinguishing subtopics or sentiments), or the general sentiment dynamics (not distinguishing subtopics), to predict the user behavior (e.g., buying a Nano).

All these results show that our method is effective to extract the dynamics of topics and the sentiments.

6. RELATED WORK

To the best of our knowledge, modeling the mixture of topics and sentiments has not been addressed in existing work. However, there are several lines of related work.

Weblogs have been attracting increasing attentions from researchers, who consider weblogs as a suitable test bed for many novel research problems and algorithms [11, 7, 6, 15, 19]. Much new research work has found applications to weblog analysis, such as community evolution [11], spatiotemporal text mining [15], opinion tracking [20, 15, 19], information propagation [7], and user behavior prediction [6].

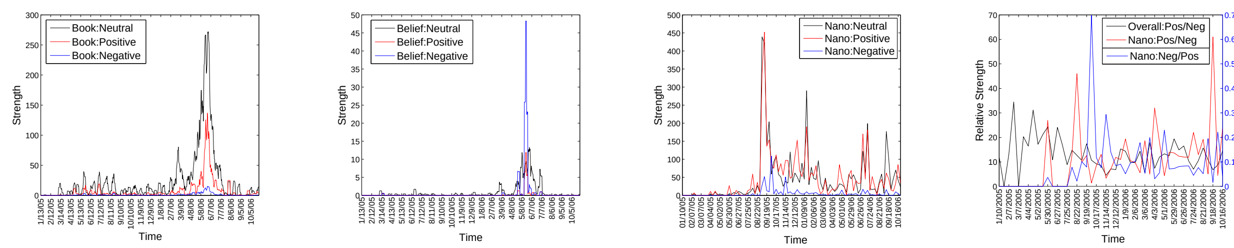
Mei and others introduced a mixture model to extract the subtopics in weblog collections, and track their distribution over time and locations [16]. Gruhl and others [7] proposed a model for information propagation and detect spikes in the diffusing topics in weblogs, and later use the burst of blog mentions to predict spikes of sales of this book in the near future [6]. However, all these models tend to ignore the sentiments in the weblogs, and only capture the general

	Neutral	Thumbs Up	Thumbs Down
Topic1 **(Movie)**	... Ron Howards selection of Tom Hanks to play Robert Langdon.	Tom Hanks stars in the movie, who can be mad at that?	But the movie might get delayed and even killed off if he loses.
	Directed by: Ron Howard Writing credits: Akiva Goldsman ...	Tom Hanks, who is my favorite movie star act the leading role.	protesting ... will lose your faith by ... watching the movie.
	After watching the movie I went online and some research on ...	Anybody is interested in it?	**... so sick of people making such a big deal about a FICTION book and movie.**
Topic2 **(Book)**	I knew this because I was once a follower of feminism.	And I'm hoping for a good book too.	**... so sick of people making such a big deal about a FICTION book and movie.**
	I remembered when i first read the book, I finished the book in two days.	Awesome book.	This controversy book cause lots conflict in west society.
	I'm reading "Da Vinci Code" now.	So still a good book to past time.	in the feeling of deeply anxious and fear, to ... read books calmly was quite difficult.

Table 6: Topic-sentiment summarization: Da Vinci Code

	TSM		Opinmind	
	Thumbs Up	**Thumbs Down**	**Thumbs Up**	**Thumbs Down**
1	(sweat) iPod Nano ok so ...	WAT IS THIS SHIT??!!	I love my iPod, I love my G5...	I hate ipod.
	Ipod Nano is a cool design, ...	ipod nanos are TOO small!!!!	I love my little black 60GB iPod	Stupid ipod out of batteries...
2	the battery is one serious	Poor battery life ...	I LOVE MY iPOD	" hate ipod " = 489..
	example of excellent relibability	...iPod's battery completely died	I love my iPod.	my iPod looked uglier...surface...
3	My new VIDEO ipod arrived!!!	fake video ipod	- I love my iPod.	i hate my ipod.
	Oh yeah! New iPod video	Watch video podcasts ...	... iPod video looks SO awesome	... microsoft ... the iPod sucks

Table 7: Topic-sentiment summarization: iPod



(a) Da Vinci Code: Book (b) Da Vinci Code: Religion (c)iPod: Nano (d)iPod: Relative

Figure 6: Topic life cycles and sentiment dynamics

description about topics. This may limit the usefulness of their results. Mishne and others instead used the temporal pattern of sentiments to predict the book sales. Opinmind [20] summarizes the weblog search results with positive and negative categories. On the other hand, researchers also use facets to categorize the latent topics in search results [8]. However, all this work ignores the correlation between topics and sentiments. This limitation is shared with other sentiment analysis work such as [18].

Sentiment classification has been a challenging topic in Natural Language Processing (see e.g., [26, 2]). The most common definition of the problem is a binary classification task of a sentence to either the positive or the negative polarity [23, 21]. Since traditional text categorization methods perform poorly on sentiment classification [23], Pang and Lee proposed a method using mincut algorithm to extract sentiments and subjective summarization for movie reviews [21]. In some recent work, the definition of sentiment classification problem is generalized into a rating scale [22]. The goal of this line of work is to improve the classification accuracy, while we aim at mining useful information (topic/sentiment models, sentiment dynamics) from weblogs. These methods do not either consider the correlation of sentiments and topics or model sentiment dynamics.

Some recent work has been aware of this limitation. Engström studied how the topic dependence influences the accuracy of sentiment classification and tried to reduce this dependence [5]. In a very recent work [4], the author proposed a topic dependent method for sentiment retrieval, which assumed that a sentence was generated from a probabilistic model consisting of both a topic language model and a sentiment language model. A similar approach could be found in [27]. Their vision of topic-sentiment dependency is similar to ours. However, they do not consider the mixture of topics in the text, while we assume that a document could cover multiple subtopics and different sentiments. Their model requires that a set of topic keywords is given by the user, while our method is more flexible, which could extract the topic models in an unsupervised/semi-supervised way with an EM algorithm. They also requires sentiment training data for every topic, or manually input sentiment keywords, while we can learn general sentiment models applicable to ad hoc topics.

Most opinion extraction work tries to find general opinions on a given topic but did not distinguish sentiments [28, 15]. Liu and others extracted product features and opinion features for a product, thus were able to provide sentiments for different features of a product. However, those product opinion features are highly dependent on the training data sets, thus are not flexible to deal with ad hoc queries and topics. The same problem is shared with [27]. They also did not provide a way to model sentiment dynamics.

There is yet another line of research in text mining, which tries to model the mixture of topics (themes) in documents [9, 1, 16, 15, 17, 12]. The mixture model we presented is along this line. However, none of this work has tried to model the sentiments associated with the topics, thus can not be applied to our problem. However, we do notice that the TSM model is a special case of some very general topic models, such as the CPLSA model [17], which mixes themes with different views (topic, sentiment) and different coverages (sentiment coverages). The generation structure in Figure 2 is also related to the general DAG structure presented in [12].

7. CONCLUSIONS

In this paper, we formally define the problem of topic-sentiment analysis and propose a new probabilistic topic-sentiment mixture model (TSM) to solve this problem. With this model, we could effectively (1) learn general sentiment models; (2) extract topic models orthogonal to sentiments, which can represent the neutral content of a subtopic; and (3) extract topic life cycles and the associated sentiment dynamics. We evaluate our model on different Weblog collections; the results show that the TSM model is effective for topic-sentiment analysis, generating more useful topic-sentiment result summaries for blog search than a state-of-the-art blog opinion search engine (Opinmind).

There are several interesting extensions to our work. In this work, we assume that the content of sentiment models is the same for all topics in a collection. It would be interesting to customize the sentiment models according to each topic and obtain different contextual views [17] of sentiments on different facets. Another interesting future direction is to further explore other applications of the TSM, such as user behavior prediction.

8. ACKNOWLEDGMENTS

We thank the three anonymous reviewers for their comments. This work is in part supported by the National Science Foundation under award numbers 0425852, 0347933, and 0428472.

9. REFERENCES

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

[2] Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of HLT-EMNLP 2005*, 2005.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statist. Soc. B*, 39:1–38, 1977.

[4] K. Eguchi and V. Lavrenko. Sentiment retrieval using generative models. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 345–354, July 2006.

[5] C. Engstrom. Topic dependence in sentiment classification. masters thesis. university of cambridge. 2004.

[6] D. Gruhl, R. Guha, R. Kumar, J. Novak, and A. Tomkins. The predictive power of online chatter. In *Proceedings of KDD '05*, pages 78–87, 2005.

[7] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *Proceedings of the 13th International Conference on World Wide Web*, pages 491–501, 2004.

[8] M. A. Hearst. Clustering versus faceted categories for information exploration. *Commun. ACM*, 49(4):59–61, 2006.

[9] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of SIGIR '99*, pages 50–57, 1999.

[10] R. Krovetz. Viewing morphology as an inference process. In *Proceedings of SIGIR '93*, pages 191–202, 1993.

[11] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. On the bursty evolution of blogspace. In *Proceedings of the 12th International Conference on World Wide Web*, pages 568–576, 2003.

[12] W. Li and A. McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 577–584, 2006.

[13] B. Liu, M. Hu, and J. Cheng. Opinion observer: analyzing and comparing opinions on the web. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 342–351, 2005.

[14] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley, 1997.

[15] Q. Mei, C. Liu, H. Su, and C. Zhai. A probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 533–542, 2006.

[16] Q. Mei and C. Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *Proceedings of KDD '05*, pages 198–207, 2005.

[17] Q. Mei and C. Zhai. A mixture model for contextual text mining. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 649–655, 2006.

[18] G. Mishne and M. de Rijke. MoodViews: Tools for blog mood analysis. In *AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW 2006)*, pages 153–154, 2006.

[19] G. Mishne and N. Glance. Predicting movie sales from blogger sentiment. In *AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW 2006)*, 2006.

[20] Opinmind. http://www.opinmind.com.

[21] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*, pages 271–278, 2004.

[22] B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, pages 115–124, 2005.

[23] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86, 2002.

[24] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–285, Feb. 1989.

[25] T. Tao and C. Zhai. Regularized estimation of mixture models for robust pseudo-relevance feedback. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 162–169, 2006.

[26] J. Wiebe, T. Wilson, and C. Cardie. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation (formerly Computers and the Humanities)*, 39, 2005.

[27] J. Yi, T. Nasukawa, R. C. Bunescu, and W. Niblack. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Proceedings of ICDM 2003*, pages 427–434, 2003.

[28] C. Zhai, A. Velivelli, and B. Yu. A cross-collection mixture model for comparative text mining. In *Proceedings of KDD '04*, pages 743–748, 2004.