Available online at www.sciencedirect.com
ScienceDirect
Expert Systems with Applications 35 (2008) 581–590
Expert Systems with Applications
www.elsevier.com/locate/eswa

Machine learning techniques for business blog search and mining

Yun Chen, Flora S. Tsai *, Kap Luk Chan

School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore

Abstract

Weblogs, or blogs, have rapidly gained in popularity over the past few years. In particular, the growth of business blogs that are written by or provide commentary on businesses and companies opens up new opportunities for developing blog-specific search and mining techniques. In this paper, we propose probabilistic models for blog search and mining using two machine learning techniques, latent semantic analysis (LSA) and probabilistic latent semantic analysis (PLSA). We implement the models in our database of business blogs, BizBlogs07, with the aim of achieving higher precision and recall. The probabilistic model is able to segment the business blogs into separate topic areas, which is useful for keywords detection on the blogosphere. Various term-weighting schemes and factor values were also studied in detail, which reveal interesting patterns in our database of business blogs. Our multi-functional business blog system is indeed found to be very different from existing blog search engines, as it aims to provide better relevance and precision of the search.

Keywords: Latent semantic analysis; Probabilistic latent semantic analysis; Weblog; Blog; Data mining

1. Introduction

After a slow start in the early 1990s, blogging rapidly gained in popularity in the mid 1990s. According to David Sifry of Technorati, over 37.3 million blogs were being tracked by Technorati in May 2006; on average, a new weblog is created every second. Because of the huge volume of existing blog posts and their free format nature, the information in the blogosphere is rather random and chaotic; effective access and retrieval techniques are needed to improve the searching quality. However, blog search engines are currently still in their infancy (Mishne, 2006). In searching for competitive intelligence, blog-specific search engines only index feeds, which are readable XML versions of the blogs, and only consist of the summary or the first few sentences of the blog entries, but not the full text; and the searching results are simply sorted by date or by "popularity" (Pikas, 2005). Moreover, blog searches have different interests than normal web searches, which track references to known entities and focus on certain concepts (Mishne & de Rijke, 2006). Hence, to increase the quality and accuracy of the search, more complex models and information retrieval techniques are required for mining the blogs.

The business world has experienced significant influence by the blogosphere. A hot topic in the blogosphere may affect a product's life cycle. An exposure of an inside story in the blogosphere may influence a company's reputation. Studies have been conducted to measure the influence of the blogosphere (Gill, 2004). There are many aspects of tracking the blogosphere to measure its influence, such as analyzing the blog threads for discovering the important bloggers (Nakajima, Tatemura, Hino, Hara, & Tanaka, 2005); focusing the topic-centric view of the blogosphere (Avesani, Cova, Hayes, & Massa, 2005); detecting the blogs growing trends (Glance et al., 2004); and tracking the propagation of discussion topics in the blogosphere (Gruhl, Guha, Liben-Nowell, & Tomkins, 2004). Although some complex algorithms have been developed to track the trends and influences of the blogosphere, other machine learning techniques can be applied in the tracking of blogs.

Focusing on mining the individual business blogs written by professionals, as well as the corporate blogs, we implemented machine learning techniques for mining content

* Corresponding author.
E-mail address: fst1@columbia.edu (F.S. Tsai).

0957-4174/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2007.07.015

from business blogs by building a multi-functional blog directory, which engaged a different approach from the mechanism of the existing blog-specific search engine. Latent semantic analysis (LSA) is proposed for mining content from blogs, and hence a search engine is built for searching blog entries more accurately, with specific measures on similarity and rankings. To broaden the usefulness of the blog search engine, probabilistic latent semantic analysis (PLSA) is applied to detect the keywords from various blog entries with respect to certain topics. This simple algorithm presents the blogosphere in terms of topics with measurable keywords, and hence it tracks the popular conversations and topics in the blogosphere.

The paper is organized as follows. Section 2 reviews the related work on blog search and mining. Section 3 describes machine learning algorithms for mining blog-related topics. Section 4 presents experimental results, and Section 5 concludes the paper.

2. Review of related work

This section reviews related work in developing blog-specific search engines and extraction of useful information from blogs.

2.1. Blog-specific search engines

Search engines have been widely used on the Internet for decades, with Google and Yahoo currently the most popular. As blogging becomes a hit in the air, blog-specific search engines are created to suit the demand. Past studies (Gill, 2004; Pikas, 2005) have examined the characteristics, effectiveness and distinctiveness of blog-specific search engines.

As of October 2006, Technorati[1] has indexed over 56 million blogs. It is believed to be tracking the largest number of links in real-time. RSS (really simple syndication) or XML feeds automatically send notification from blogs to Technorati quickly, and the thousands of updates per hour that occur in the blogosphere are tracked by Technorati. Technically, Technorati supports open microformat standards, and indexes and searches posts tagged with rel-tag, using the full boolean search technique on the RSS/XML files. This full boolean search functionality provides AND, OR and NOT functionality, which narrows down the searching results, and has made tagging more valuable. In order to provide the users with more options, search results of synonymous queries are listed. For example, when "car" is the search query, there is an option of refining the results on "auto", "automotive", "automobile", "autos", "life", "vehicles", "work".

Similarly, many other blog-specific search engines, such as Bloglines[2], Feedster[3], and BlogPulse[4], index and search the RSS/XML feeds of blogs, using the boolean search. However, the ranking of the results are not in the order of relevance, but rather in the time of posting, the "popularity" (number of links to the blog), or the update frequency. More importantly, the existing blog search engines do not necessarily use complex models to achieve better results, due to the constraints of the real-time nature of blogs and its extremely rapid speed of update in the blogosphere. Thus, the quality of blog search results can be improved.

2.2. Extraction of useful information from blogs

Current blog text analysis focuses on extracting useful information from blog entry collections, and determining certain trends in the blogosphere. Natural Language Processing (NLP) algorithms have been used to determine the most important keywords and proper names within a certain time period from thousands of active blogs, which can automatically discover trends across blogs, as well as detect key persons, phrases and paragraphs (Glance et al., 2004). A study on the propagation of discussion topics through the social network in the blogosphere developed algorithms to detect the long-term and short-term topics and keywords, which were then validated with real blog entry collections (Gruhl et al., 2004). On evaluating the suitable methods of ranking term significance in an evolving RSS feed corpus, three statistical feature selection methods were implemented: χ^2, mutual information (MI) and information gain (I), and the conclusion was that χ^2 method seems to be the best among all, but full human classification exercise would be required to further evaluate such a method (Prabowo & Thelwall, 2006). A probabilistic approach based on PLSA was proposed in Mei, Liu, Su, and Zhai (2006) to extract common themes from blogs, and also generate the theme life cycle for each given location and the theme snapshots for each given time period. This study illustrates that the PLSA-based approach using probabilistic mixture models can be effectively used for viewing spatiotemporal life cycle patterns of blogs.

Our work differs from existing studies in two respects: (1) We focus on business and corporate blog entries which may contain less extraneous and more useful information than personal blogs, potentially resulting in higher quality search results and (2) we have combined a blog search engine with topic and keyword extraction, and use probabilistic models to extract popular keywords for each topic.

3. Machine learning techniques for blog search and mining

3.1. Latent semantic analysis

Latent semantic analysis (LSA), also known as latent semantic indexing (LSI), is an automatic indexing and retrieval technique, which was initially designed for improved detection of relevant documents on the basis of

[1] http://www.technorati.com.
[2] http://www.bloglines.com.
[3] http://www.feedster.com.
[4] http://www.blogpulse.com.

search queries (Deerwester, Dumais, Landauer, Furnas, & Harshman, 1990), but also has other applications in web page classification (Chen & Hsieh, 2006) and topic detection in source code (Kuhn, Ducasse, & Girba, 2007). The implicit higher-order structure of terms with documents is employed in the technique, hence the term "semantic structure". It applies the singular value decomposition (SVD) method to project the document vectors into a lower dimensional subspace, measures the semantic similarity, reflects the major associative pattern in the indexing data, and ignores the less representative information.

LSA is a dimensionality reduction technique addressing two issues from information retrieval, broadly named as synonymy and polysemy. The vector space analysis, singular-value decomposition (SVD), determines the correlation between the occurrence of terms and documents in the term by document matrix, and hence evaluates the "real" association of terms and documents. Also, some pattern of the occurrence of some words provides evidence of the likely occurrence of other words. Hence, terms with similar meanings of the search query are emphasized and the synonymy problem is almost resolved. At the same time, the co-relational structure analysis on predicting the likely occurrence of other terms based on the pattern of the occurrence of the query tries to anticipate the searcher's desired knowledge, and hence it reduces the effects caused by the polysemous issue.

3.1.1. Term-document matrix (TDM) generation

In the LSA process, the first step is to preprocess the document collection, and then create the term-document matrix (TDM), which is the conversion of the real-life text into the numerical representation as the input for LSA algorithm (Berry, Dumais, & O'Brien, 1995). Firstly, all the formatting (i.e. capitalization and punctuation) and extraneous markup (i.e. dateline), are stripped from all articles, forming a term list from the articles. A stop words list is used as a filter to eliminate a large number of the commonly used English words, such as "the", "to", "have", from the obtained term list, hence reduce much noise in the collection. Stemming is next applied to the reduced term list, further removing common endings from words. Then a TDM could be generated by arranging the list of all content words along the vertical axis, and a similar list of all documents along the horizontal axis, in any order. The TDM is then filled in with the frequency of occurrence of each word in each corresponding document. Since each document only contains a portion of words of the whole content word list, the resultant matrix is very sparse with the existence of many zeros. Although this simple version of TDM can give acceptable results, the performances can be further improved by applying the term-weighting schemes and document length normalization techniques.

Under the application of term-weighting schemes, the values of a cell in the raw TDM can be transformed so that the frequency of each term in a document is recalculated. This transformation include two components: a global weighting and a local weighting (Dumais, 1991). Each element, a_{ij}, of a TDM can be expressed as

$$a_{ij} = l_{ij} g_i n_{ij} \tag{1}$$

where l_{ij} is a local factor that measures the importance of term i in document j, g_i is a global factor that measures the importance of term i in the entire collection, and n_{ij} is a normalization factor.

Studies are done on examining the efficiency of various term-weighting schemes (Dumais, 1991; Kolda, 1997; Zeimpekis & Gallopoulos, 2006). Two global weighting and three local weighting schemes, which are found to give relatively high efficiency, are listed in Table 1. Symbol f_{ij} denotes term frequency, i.e., the number of times term i appears in document j. The probability function is defined as

$$p_{ij} = \frac{f_{ij}}{\sum_k f_{ik}} \tag{2}$$

and the binary function is defined as

$$b(f_{ij}) = \begin{cases} 1, & \text{if } f_{ij} \neq 0 \\ 0, & \text{if } f_{ij} = 0 \end{cases} \tag{3}$$

Normalization penalizes the term weights for a document in accordance with its length to fairly retrieve documents of all lengths (Singhal, Buckley, & Mitra, 1996). The improved pivoted cosine normalization scheme, which reduces the gap between the relevance and the retrieval probability, was proposed and implemented in a Matlab toolbox for term-document matrix generation (Zeimpekis & Gallopoulos, 2006), and it is denoted as

$$\left(\sum_j (g_i l_{ij})^2 \right)^{-1/2} \tag{4}$$

This formula replaces the cosine normalization and is widely used in the information retrieval domain.

Table 1
Term weighting schemes of high efficiency (Zeimpekis & Gallopoulos, 2006)

Symbol	Name	Type
	Local term-weighting (l_{ij})	
tf	Term frequency	f_{ij}
lo	Logarithmic	$\log_2(1+f_{ij})$
al	Alternate log	$b(f_{ij})(1+\log_2 f_{ij})$
	Global term-weighting (g_{ij})	
EN	Entropy	$1+\sum_j (p_{ij}\log_2(p_{ij}))/\log_2 n$
IDF	Inverse document frequency	$\log_2\left(n \Big/ \sum_j b(f_{ij})\right)$

3.1.2. Singular value decomposition (SVD)

The singular value decomposition (SVD) is commonly applied in the solution of unconstrained linear least squares problems, matrix rank estimation, and canonical correlation analysis (Yu, Cuadrado, Ceglowski, & Payne, 2002). The singular value decomposition of the $m \times n$ matrix A (where $m \geqslant n$, and rank $(A) = r$), denoted by SVD (A), is defined as

$$A = U \sum V^{\mathrm{T}} \tag{5}$$

where $U^{\mathrm{T}}U = V^{\mathrm{T}}V = I_n$ and $\sum = \mathrm{diag}(\sigma_1, \ldots, \sigma_n), \sigma_i > 0$, for $1 \leqslant i \leqslant r$, $\sigma_j = 0$, for $j \geqslant r+1$. In LSA, matrix A is the term-document matrix, representing the occurrence of each term in the documents of the corpus, denoted as $A = [a_{ij}]$. By applying the Eckart–Young theorem, A_k is an equivalent to the approximation of original matrix A, with k factors or k largest singular triplets, denoted as

$$A_k = \sum_{i=1}^{k} u_i \cdot \sigma_i \cdot v_i^{\mathrm{T}} \tag{6}$$

Each of the SVD components is interpreted in meaning context within LSA (refer to Table 2). The choice of the value of k, which is the reduced dimensional representation, is an interesting problem, and it is difficult to find the best solution. Indeed, the best value for k is usually determined through experiments. By examining the relation between performances and the number of dimensions, it is deduced between 70 and 100 dimensions, the performance of search is the best (Deerwester et al., 1990).

In short, the SVD method projects a typical large term-document matrix, which may have tens of thousands of dimensions, into a much smaller set of dimensions, preserving as much information as possible about the relative distances between the document vectors. During the collapse of the original term-document matrix into its reduced approximation, the "noise" information is lost, but the content words are superimposed on one another, revealing latent similarity in the document collection (Yu et al., 2002).

3.1.3. Cosine similarity

As LSA is a vector space model in information retrieval, the standard cosine similarity is used to measure the similarity between the search query and the documents (Berry & Browne, 2005). The cosine similarity measures the cosine of the angle between the query vector and the document vector. If d_j is defined as the jth document vector (i.e. the jth column of the reduced singular value of the $m \times n$ term-document matrix), then the cosine between the query vector $q = (q_1, q_2, \ldots, q_m)^{\mathrm{T}}$ and the document vector is defined as

$$\cos\theta_j = \frac{d_j^{\mathrm{T}} q}{\|d_j\|_2 \|q\|_2} = \frac{\sum_{i=1}^{m} d_{ij} q_i}{\sqrt{\sum_{i=1}^{m} d_{ij}^2} \sqrt{\sum_{i=1}^{m} q_i^2}} \tag{7}$$

for $j = 1, 2, \ldots, n$, where n is the total number of documents in the search. It is common that the query vector and document vector are sparse, containing many zeros, hence is it not excessive to compute the inner product $d_j^{\mathrm{T}} q$ and the norms $\|d_j\|_2$ and $\|q\|_2$.

The value of the cosine ranges from -1 to 1, hence the cosine similarity usually has a threshold value. Up to the design by the developer, and the threshold should be positive, where the higher the threshold value, the smaller number of documents retrieved, as fewer document vectors are closer to the query vector.

Table 2
SVD components within LSA context (Berry et al., 1995)

SVD components	Interpretation within LSA
A_k	Best rank-k approximation to A
U	Term vector
Σ	Singular values
V	Document vectors
m	Number of terms
n	Number of documents
k	Number of factors
r	Rank of A

3.2. Probabilistic latent semantic analysis model for blog mining

Probabilistic latent semantic analysis (PLSA) (Hofmann, 1999) is based on a generative probabilistic model that stems from a statistical approach to LSA (Deerwester et al., 1990). PLSA is able to capture the polysemy and synonymy in text for applications in the information retrieval domain. Similar to LSA, PLSA uses a term-document matrix which describes patterns of term (word) distribution across a set of documents (blog entries). By implementing PLSA, topics are generated from the blog entries, where each topic produces a list of word usage, using the maximum likelihood estimation method, the expectation maximization (EM) algorithm.

The starting point for PLSA is the *aspect model* (Hofmann, 1999). The aspect model is a latent variable model for co-occurrence data associating an unobserved class variable $z_k \in \{z_1, \ldots, z_k\}$ with each observation, an observation being the occurrence of a keyword in a particular blog entry. There are three probabilities used in PLSA:

(1) $P(b_i)$ denotes the probability that a keyword occurrence will be observed in a particular blog entry b_i,
(2) $P(w_j|z_k)$ denotes the class-conditional probability of a specific keyword conditioned on the unobserved class variable z_k,
(3) $P(z_k|d_i)$ denotes a blog-specific probability distribution over the latent variable space.

In the collection, the probability of each blog and the probability of each keyword are known, while the probability of an aspect given a blog and the probability of a keyword given an aspect are unknown. By using the above

three probabilities and conditions, three fundamental schemes are implemented:

(1) select a blog entry b_i with probability $P(b_i)$,
(2) pick a latent class z_k with probability $P(z_k|b_i)$,
(3) generate a keyword w_j with probability $P(w_j|z_k)$.

As a result, a joint probability model is obtained in asymmetric parameterization:

$$P(b_i, w_j) = P(b_i)P(w_j|b_i) \tag{8}$$

$$P(w_j|b_i) = \sum_{k=1}^{K} P(w_j|z_k)P(z_k|b_i) \tag{9}$$

After the aspect model is generated, the model is fitted using the EM algorithm. The EM algorithm involves two steps, namely the expectation (E) step and the maximization (M) step. The E-step computes the posterior probability for the latent variable by applying Bayes' formula, so the parameterization of joint probability model is obtained as

$$P(z_k|b_i, w_j) = \frac{P(w_j|z_k)P(z_k|b_i)}{\sum_{l=1}^{K} P(w_j|z_l)P(z_l|b_i)} \tag{10}$$

The M-step updates the parameters based on the expected complete data log-likelihood depending on the posterior probability resulted from the E-step. Hence, the M-step re-estimates the following two probabilities:

$$P(w_j|z_k) = \frac{\sum_{i=1}^{N} n(b_i, w_j)P(z_k|b_i, w_j)}{\sum_{m=1}^{M}\sum_{i=1}^{N} n(b_i, w_m)P(z_k|b_i, w_m)} \tag{11}$$

$$P(z_k|b_i) = \frac{\sum_{j=1}^{M} n(b_i, w_j)P(z_k|b_i, w_j)}{n(b_i)} \tag{12}$$

The EM iteration is continued to increase the likelihood function until the specific conditions are met and the program is terminated. These conditions can be a convergence condition, or a cut-off point, which is specified for reaching a local maximum, rather than a global maximum.

In short, the PLSA model selects the model parameter values that maximize the probability of the observed data, and returns the relevant probability distributions by applying the EM algorithm. Word usage analysis with the aspect model is a common application of the aspect model. Based on the preprocessed term-document matrix, the blogs are then classified onto different aspects or topics. For each aspect, the keyword usage, such as the probable words in the class-conditional distribution $P(w_j|z_k)$, is determined. Empirical results indicate the advantages of PLSA in reducing perplexity, and high performance of precision and recall in information retrieval (Hofmann, 1999).

4. Experimental results

We have created a business blog data set, built a blog search system using the latent semantic analysis model, and applied the probabilistic model for blog mining on our data set of business blogs. The blog search system demonstrates the ranking of business blog entries by similarity measures, and allows for full-text searching in different categories. We extract the most relevant categories and show the topics extracted for each category. Experiments show that the probabilistic model can reveal interesting patterns in the underlying topics for our data set of business blogs.

4.1. Data corpus

One thousand two hundred and sixty nine business blog entries were collected in two phases. First, a small number of blog entries are manually collected from various CEOs's blog sites and business blog sites. Meaningful blog entries from these blog sites were extracted and stored into our database. Each blog entry is saved as a text file in its corresponding category, for further text preprocessing. For the preprocessing of the blog data, we performed lexical analysis by removing stopwords and stemming using the Porter stemmer (Porter, 1980). The text files are then used as the input for the Text to Matrix Generator (TMG) (Zeimpekis & Gallopoulos, 2006) to generate the term-document matrix for input to the blog search and mining system. The overview of the system is shown in Fig. 1.

There are a total of 86 companies represented in the blog entries, and Table 3 summarizes the top companies in the BizBlogs07 data corpus, and Table 4 lists the fields in the database.

We then categorize the 1269 blog entries into four categories based on the contents or the main description of the blog: Company, Finance, Marketing, and Product. The Company category deals with news or other information specific to corporations, organizations, or businesses. The Finance category relates to financing, loans, credit information. The Marketing category deals with marketing, sales, and advertising strategies for companies. Finally, the Product category describes the blog entries on specific company products, such as reviews, descriptions, and other product-related news. Table 5 summarizes the number of blog entries in each category.

4.1.1. Blog search system

We implemented a blog search system for our business blog data. We used the LSA model (Deerwester et al., 1990) for constructing the search system, as LSA is able to consider blog entries with similar words which are semantically close, and calculate a similarity measure based on documents that are semantically similar to the query terms.

In this system, a user can select the type and the category, and enter a word or a phrase as the query to search for blog entries matching the query. The results are ranked in the order of similarity. The title of each searched blog entry is then displayed in order of similarity. Clicking on the title results in a full text of the blog entry, as well as the original hyperlink.

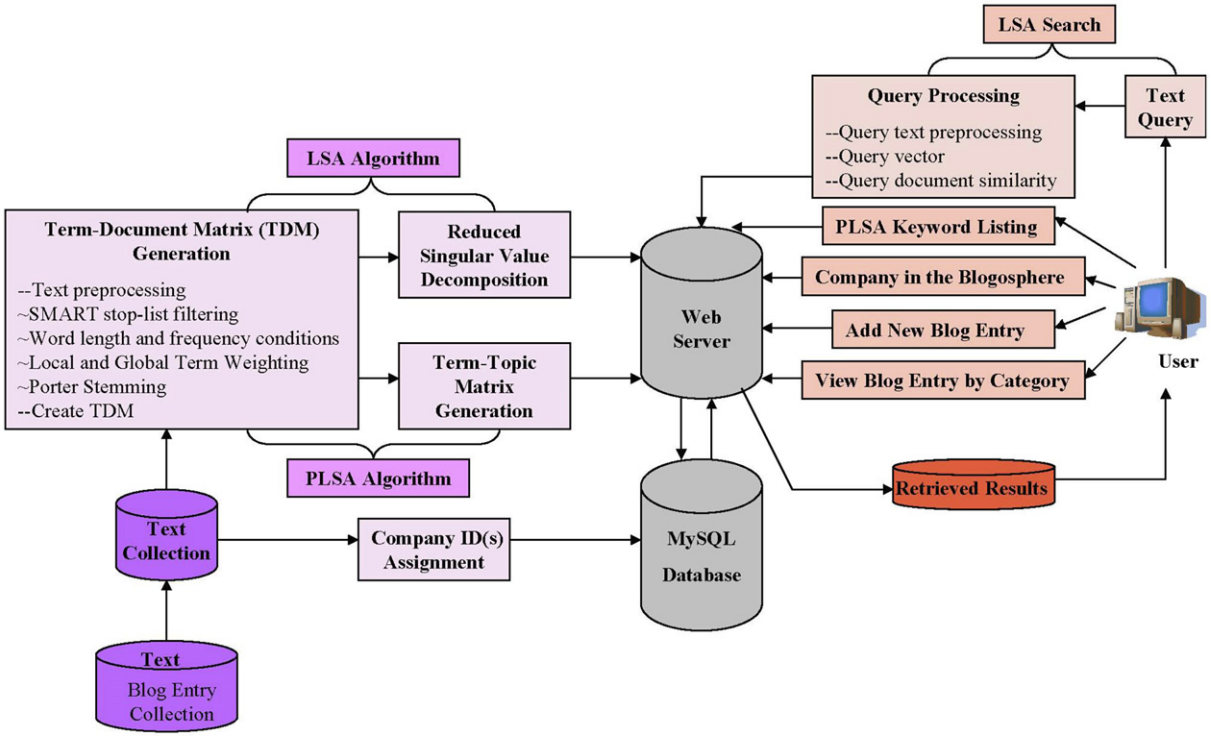

Fig. 1. Overview of business blog directory system.

Table 3
List of top companies

Company
Microsoft
eBay
Samsung
Dell
Amazon
Sony
Google
Apple
Palm
Yahoo

Table 4
Database schema for blog entry

Field	Type
ID	int(11)
Title	Text
Author	Text
Publish_Date	Date
URL	Text
Content	Text
Category_ID	int(11)
Type_ID	int(11)
Company_ID1	int(11)
Company_ID2	int(11)

4.2. Determining the efficiency of term-weighting schemes in LSA search

Although the performance of various term-weighting schemes have been examined by some researchers, there are limitations in the testing. The testing data consist of standard text collections, which are designed for information retrieval. However, there are lots of free-format text information available, such as blogs, which can be searched by machine learning techniques. In this experiment, the objective is to measure the efficiency of the LSA search for various term-weighting schemes applied on the blog entries collection, which is very random and dissimilar in terms of the style of writing and language.

Table 5
Categories for the BizBlogs07 data corpus

Categories	Number of entries
Product	387
Company	265
Finance	348
Marketing	269

The local weighting schemes tested were alternate log (*al*), term frequency (*tf*) and logarithmic (*lo*). The global weighting schemes chosen were inverse document frequency (*IDF*) and entropy (*EN*). These are the term-weighting schemes that were tested to give better performance from previous researches. There are totally six combinations of the term-weighting schemes. To measure the precision and recall of each of the search results, the queries "product", "company", "marketing" and "finance" are searched in all categories. This is based on the face that there are four categories in the data collection: product, company, marketing and finance. When searching for "marketing" in all categories, those retrieved blog entries

belong to the marketing category are relevant to the search, based on the assumption that the user trusts the categorization of the collection.

From the testing on each category, the consistency of the ordering of precisions of the categories is observed, such that the category Finance has the highest precision, followed by Marketing and Product, and Company is the category that always has the lowest precision of the search results. It is also observed that the ordering of recalls of the categories is consistent, such that the category Marketing has the highest recall, followed by Finance and Company, while Product has the lowest recall of the search results. This proves the fact that applying different term-weighting schemes will only affect the individual precision and recall of each category, but will not change the overall ordering of precision and recall of all categories. Besides, the ranking of the average similarity values for the testing term-weighting scheme combinations shows no variation among all categories. The average testing results are summarized in Table 6.

The difference in percentage of either precision or recall is not much for various term-weighting scheme combinations. However, the amount of data that exists may exceed hundreds of thousands documents. Hence, a 1% difference of the precision of the search results may cause much time and inconvenience for the user to find the relevant results. Therefore, it is suggested to use the term-weighting scheme combination that gives the highest precision with a corresponding high or moderate recall and a good measure of the cosine similarity for the search system, hence *tf–EN* is chosen for LSA search in this system.

4.3. Determining the k value in LSA search

Dimensionality reduction of term-document matrices (TDMs) and hence the generation of the reduced singular values of the TDMs will reduce the noise in text categorization and the computational complexity of cluster creation, and produces the best statistical approximation to the original vector space model, representing the latent concepts. The reduction in dimension is determined by the k values. Six different values of k are tested to generate the singular values with the term-weighting schemes *tf–EN* applied consistently. The precision and recall are measured in the same manner as in the previous section. The LSA search results are evaluated and summarized in Table 7.

Table 6
Summary of performance measurement of term weighting schemes

Ranking	Average similarity value	Average precision (%)	Average recall (%)
1	0.547721 (*tf–EN*)	50.43 (*tf–EN*)	37.15 (*al–EN*)
2	0.474572 (*tf–IDF*)	50.38 (*al–IDF*)	36.78 (*lo–IDF*)
3	0.382565 (*al–EN*)	49.16 (*lo–IDF*)	35.62 (*tf–EN*)
4	0.334056 (*lo–EN*)	48.87 (*tf–IDF*)	35.54 (*lo–EN*)
5	0.307074 (*al–IDF*)	47.03 (*al–EN*)	34.67 (*al–IDF*)
6	0.268459 (*lo–IDF*)	44.67 (*lo–EN*)	33.55 (*tf–IDF*)

Table 7
Summary of performance measurement for different k values

Ranking	Average similarity value	Average precision(%)	Average recall (%)
1	0.557135 ($k = 200$)	50.43 ($k = 100$)	91.80 ($k = 2$)
2	0.555188 ($k = 150$)	49.38 ($k = 200$)	53.16 ($k = 10$)
3	0.547721 ($k = 200$)	49.15 ($k = 150$)	40.32 ($k = 50$)
4	0.511116 ($k = 50$)	44.52 ($k = 50$)	35.62 ($k = 100$)
5	0.289833 ($k = 2$)	37.87 ($k = 2$)	27.14 ($k = 200$)
6	0.143855 ($k = 10$)	30.43 ($k = 10$)	20.93 ($k = 150$)

Looking at the similarity of the search results, the higher the value of k, the higher the value of the similarity matching the query to the first document retrieved. There is no variation in this linear relationship of the similarity and k values. In measuring the recall, there is a trend that the smaller value of k gives a higher record of recall, except for $k = 200$ which gives a higher recall than for $k = 150$. Lastly, the highest precision is achieved when $k = 100$, which is confirmed to the finding by Dumais (1991). For $k = 2$, the search results have the very high recall (i.e., average recall of 91.80%), but the results are not very relevant to the search query with an average precision of 37.87%, and the first retrieved document is quite dissimilar to the search query, with the average highest similarity of 0.143855. This explains the fact that too much reduction in the dimension of the vector space will remove essentially important information, leaving too little useful information in the latent model. Therefore, the system just simply retrieves as many documents as possible from the collection, which contain many irrelevant information. Because we aim to develop a search system that produces high relevancy of the search results, the value of k is chosen to be 100 in the application, and this value indeed gives a moderate recall in the search.

4.4. Verifying precision–recall relationship

There is an inverse relationship between precision and recall of the search efficiency. Two experiments were carried out to verify this theory and examine the performance of LSA search. By changing the query-document similarity cut-off value, the number of documents retrieved will be different; hence the precision and the relative recall will be different. The similarity cut-off values are chosen as 0.040, 0.055, 0.070, 0.090 and 0.100. The precision–recall curve can be plotted using these pair of precision and recall at each similarity cut-off. The same search query, "company" is used consistently throughout the experiments to search relevant blog entries in all categories. The results of the experiments for testing different term-weighting and k values are detailed in Figs. 2 and 3.

It is observed that within the range of similarity cut-off from 0.040 to 0.100, there is an inverse relationship between precision and recall for all the 6 term-weighting schemes combinations tested: the relative recall decreases as the precision increases. By lowering the similarity cut-off, more

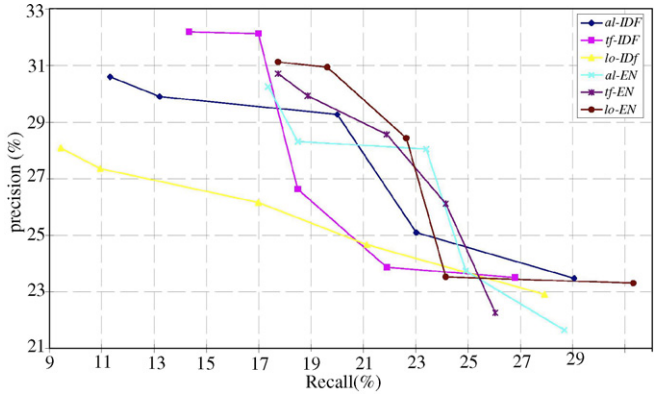

Fig. 2. Precision–recall of various term weighting.

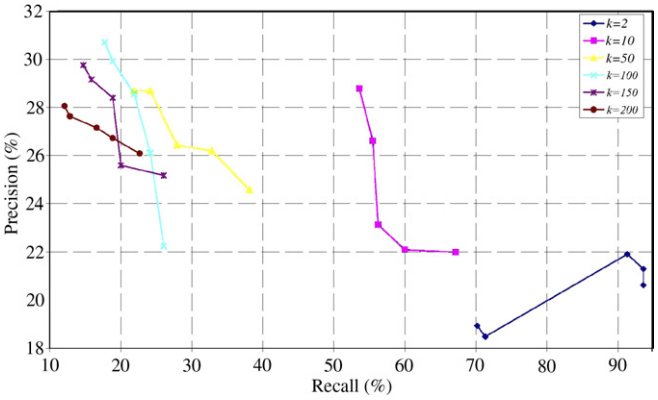

Fig. 3. Precision–recall of different k values.

documents are retrieved, which include more relevant documents; hence the higher recall percentage, but the lower precision percentage. Therefore, the recall is highest and the relative precision is lowest at similarity cut-off at 0.040, and reverse at the similarity cut-off at 0.100. The plots are concentrated at a range of precision (21–33%) and recall (9–32%). Among all the plots, the curves of *tf–EN* and *lo–IDF* are relative smooth, showing consistent and gradual changes. However, the curve of *lo–IDF* tends to be at the bottom, and covers larger range of recall than that of *tf–EN*, which tends to be at the top of the graph. *tf–EN* will generates results of higher precision, with relatively constant recall; this term-weighting scheme combination is confirmed to be applied in the system.

Fig. 3 shows that, except for $k = 2$, the inverse relationship between precision and recall are valid for other values of k. Also, the plots cover a larger range of precision and recall than that in Fig. 2; this proves that changing the value of k significantly affects the relevance of the search. The plot for $k = 2$ is at the right bottom of the graph, and shows no constant relationship between precision and recall. At $k = 2$, there is too much reduction in dimensions of the vector model, hence variation from the precision–recall relationship is observed. Plot of $k = 10$ tends to show good precision and recall at high similarity cut-off, such as 0.100, but gives poor precision at relatively low similarity cut-off values, such as 0.070 and below. The curves for $k = 100$ and $k = 200$ are rather smooth. However, the curve for $k = 100$ is at a higher position and to the right of that for $k = 200$ in the graph; therefore at $k = 100$, higher precision and recall will be obtained in the search than at $k = 200$. Thus, we chose a value of k to be 100 for generating the singular values of the vector models.

4.5. Results from PLSA keyword listing

The first 10 keywords with the highest probability to the corresponding topic from each category (Company, Finance, Marketing, and Product) of the entire blog entry collection are listed in Tables 8–11.

These keyword listings identify the popular topics in the blogosphere. It is worthy to mention the interesting findings from the PLSA keyword listing, when searching for four topics in all categories. The four topics from this search are actually matched to the four categories in the database, as shown in Table 5. This reflects that the categorization of the blog entry collection is tenable, and this categorization can be applied in business blogs in large. By looking at the various topics listed, we are able to see that the probabilistic approach is able to list important keywords of each topic in a quantitative fashion. The keywords listed can relate back to the original topics. For example, the keywords detected in the Product topic features items such as

Table 8
List of keywords for Topic 1 (Product)

Keyword	Probability
Mobile	0.01320409
Battery	0.00845365
Device	0.00826390
Phone	0.00825595
Window	0.00673951
Tablet	0.00650617
Umpc	0.00603427
Samsung	0.00555756
Keyboard	0.00475929
Apple	0.00474569

Table 9
List of keywords for Topic 2 (Company)

Keyword	Probability
Blog	0.01099289
ebay	0.00924013
Amazon	0.00734261
Google	0.00546131
Web	0.00521787
Develop	0.00512831
Api	0.00501918
Site	0.00478636
Search	0.00449269
Product	0.00425156

Table 10
List of keywords for Topic 3 (Finance)

Keyword	Probability
Save	0.01575988
Money	0.01410636
Debt	0.00743223
Year	0.00680512
Financ	0.00625563
Financi	0.00621918
Credit	0.00593993
Card	0.00591865
College	0.00591530
Invest	0.00570176

Table 11
List of keywords for Topic 4 (Marketing)

Keyword	Probability
Market	0.01182965
Company	0.00715767
Custom	0.00650473
Busi	0.00578797
Firm	0.00473713
Advertise	0.00384597
Brand	0.00370062
Product	0.00358449
Corpor	0.00328202
Client	0.00324660

mobile products, batteries, phones, and umpc (ultra mobile PC). In this way, it is possible to list popular keywords and track the hot topics in the blogosphere.

The power of PLSA in business blog applications include the ability to automatically detect terms and keywords related to business and corporate blog trends in product, marketing, and finance matters. By presenting blogs with measurable keywords, we can improve our understanding of business issues in terms of distribution and trends of current conversations and events. This has implications for companies wishing to monitor real-time business trends present in weblogs or other related documents.

5. Conclusions

This paper presents results using probabilistic and latent semantic models for search and analysis of business blogs. To our knowledge, is the first such study focusing on business blogs. We have created a business blog data corpus for this study, and categorized the data set into four classes. A multi-functional business blog directory has been successfully developed in searching and mining the business blogs. The system implements a different approach from the existing blog search engines, and is able to provide a higher relevance of the search. Various term-weighting schemes and factor values for similarity search have been studied in detail. The employed measures have proved to be appropriate for improving the precision and recall for our business blog data set. Our experiments on our data set of business blogs demonstrate how our probabilistic blog model can present the blogosphere in terms of topics with measurable keywords, hence tracking popular conversations and topics in the blogosphere. We hope that this work will contribute to the growing need and importance for search and mining of business blogs.

Potential applications of this stream of research may include automatically monitoring and identifying trends in business blogs. This can have some significance for organizations and companies wishing to monitor real-time trends in business marketing, finance, and product information present in weblog conversations and the blogosphere.

References

Avesani, P., Cova, M., Hayes, C., & Massa, P. (2005). Learning contextualised weblog topics. In *Proceedings of the WWW '05 workshop on the weblogging ecosystem: aggregation, analysis and dynamics*.

Berry, M., Dumais, S., & O'Brien, G. (1995). Using linear algebra for intelligent information retrieval. *SIAM Review, 37*(4), 573–595.

Berry, M. W., & Browne, M. (2005). *Understanding search engines: Mathematical modeling and text retrieval* (2nd ed.). Philadelphia, PA: SIAM.

Chen, R.-C., & Hsieh, C.-H. (2006). Web page classification based on a support vector machine using a weighted vote schema. *Expert Systems with Applications, 31*(2), 427–435.

Deerwester, S., Dumais, S., Landauer, T., Furnas, G., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science, 41*(6), 391–407.

Dumais, S. T. (1991). Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments and Computers, 23*(2), 229–236.

Gill, K. E. (2004). How can we measure the influence of the blogosphere? In *Proceedings of the WWW '04 workshop on the weblogging ecosystem: aggregation, analysis and dynamics*.

Glance, N. S. Hurst, M., & Tomokiyo, T. (2004). BlogPulse: automated trend discovery for weblogs. In *Proceedings of the WWW '04 workshop on the weblogging ecosystem: aggregation, analysis and dynamics*.

Gruhl, D., Guha, R., Liben-Nowell, D., & Tomkins, A. (2004) Information diffusion through blogspace. In *Proceedings of the WWW '04 workshop on the weblogging ecosystem: aggregation, analysis and dynamics*.

Kolda, T. G. (1997). Limited-memory matrix with applications. In Technical Report CS-TR-3806. PhD Thesis, University of Maryland, College Park, USA.

Kuhn, A., Ducasse, S., & Girba, T. (2007). Semantic clustering: Identifying topics in source code. *Information and Software Technology*.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval* (pp. 50–57).

Mei, Q., Liu, C., Su, H., & Zhai, C. (2006). A Probabilistic approach to spatiotemporal theme pattern mining on weblogs. In *Proceedings of the WWW '06 workshop on the weblogging ecosystem: aggregation, analysis and dynamics*.

Mishne, G., & de Rijke, M. (2006). A Study of blog search. In *Proceedings of the ECIR '06*.

Mishne, G. (2006). Information access challenges in the blogspace. In *Proceedings of the IIIA-2006: International workshop on intelligent information access, Helsinki, Finland.*

Nakajima, S., Tatemura, J., Hino,Y., Hara,Y., & Tanaka, K. (2005). Discovering important bloggers based on analyzing blog threads. In *Proceedings of the WWW '05 workshop on the weblogging ecosystem: aggregation, analysis and dynamics.*

Pikas, C. K. (2005). Blog searching for competitive intelligence, brand image, and reputation management. *Online, 29*(4), 16–21.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*(3), 130–137.

Prabowo, R., & Thelwall, M. (2006). A comparison of feature selection methods for an evolving RSS feed corpus. *Information Processing and Management, 42*(6), 1491–1512.

Singhal, A., Buckley, C., & Mitra, M. (1996). Pivoted document length normalization. In: *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 21–29).

Yu, C., Cuadrado, J., Ceglowski, M., & Payne, J. S. (2002). Patterns in unstructured data: discovery, aggregation, and visualization. In Presentation to the Andrew W. Mellon Foundation.

Zeimpekis, D., & Gallopoulos, E. (2006). TMG: A MATLAB Toolbox for generating term-document matrices from text collections. In *Grouping multidimensional data: Recent advances in clustering* (pp. 187–210).