A Novel Judge Mechanism to Enhance the Performance of Google Blog Search

Lin-Chih Chen

Department of Information Management, National Dong Hwa University, Taiwan
lcchen@mail.ndhu.edu.tw

Abstract

In recent years, the online blogging community is growing bigger as the social network service. When it is growing, the blog posts are increasing day by day. Generally speaking, people were using the blog search engines to search and recommend potentially interesting blog posts. When people search from the blog search engines, they were faced with two major problems: synonymy (two different terms with the same meaning) and polysemy (a term with different meanings). In this paper, we use two semantic analysis methods, Latent Semantic Indexing (LSI) and Probabilistic LSI (PLSI), to solve these two problems. LSI uses singular value decomposition as the fundamental method to capture the synonymous relationship between terms. PLSI uses the Expectation-Maximization algorithm for parameter estimation to additionally deal with the problem of polysemy. Although PLSI can gracefully deal with these two semantic problems, it needs a huge computing time. To solve the problem of computing time, in this paper, we propose a novel termination mechanism to dynamically determine the required number of iterations for PLSI. According to the experiment results, the result derived from our mechanism can not only deal with these two semantic problems but also reach a cost-effective solution.

Keywords: Blog posts, Google blog search, Web 2.0, Expectation-maximization algorithm, Semantic problems

1 Introduction

As information technology advances rapidly, hosts with Internet access capability are becoming increasingly popular and a growing number of applications and services on the web. From the early start, web 1.0 is the first generation of the web which according to Tim Berners-Lee. In this generation, web includes only static web content instead of dynamic user-generated content. It is only one way of pushing web content to many people via websites visited and emails transmitted.

The key evolution from web 1.0 to web 2.0 is to allow the users to build an interactive, collaborative, and user-generated content. According to Best [3], the main characteristics of web 2.0 compared to web 1.0 are the following: rich user experience, user participation, dynamic content, metadata, web standards, and scalability. There are many famous examples of web 2.0, including social networking sites, blogs, wiki, and stream media video sites.

One popular web 2.0 application is the blog, a hybrid between a diary and an online journal, characterized by chronological ordering of information [47]. A user can publish a blog post and other users can read and comment on it. There are two major advantages for creating a blog: (1) it adds some more depth to the blog posts; (2) it builds community and reader loyalty [46].

From the early ages to our recent years, the exponential rise in the number of blogs from thousands in the late 1990s to hundreds of millions in 2015 [52, 54] has created a need for effective access and retrieval services. Today, there are many blog search engines available on the web, such as Google Blog Search, Amatomu, Bloglines, BlogScope, IceRocket, Munax, Regator, and Technorati to help the users find useful blog posts quickly from such huge blogs.

Blog search engines have become more and more sophisticated, helping users find appropriate blog sites that offer frequently updated content on pretty much any topic users can possibly think of. However, two additional problems with most traditional blog search engines are the low coverage (or called synonymy) and the lack of disambiguation ability (or called polysemy) [45, 58, 61]. In some cases two relevant terms never occur with each other, and they will not be found by the traditional blog search engines. In other cases a term may have several different meanings in the search results. For example, the search results for an “apple” query (by Regator) contain an abundance of pages about a computer company and the first search result that related to a kind of fruit is located at about the 30th place. A user must refine his/her query in order to find relevant blog posts; this process can be long and annoying especially for an inexperienced user.

*Corresponding Author: Lin-Chih Chen; E-mail: lcchen@mail.ndhu.edu.tw
DOI: 10.3966/160792642018081904002

In this paper, we use two semantic analysis methods, Latent Semantic Indexing (LSI) [11] and Probabilistic LSI (PLSI) [23], to deal with the problems of synonymy and polysemy as mentioned above for the study of blog search engine. LSI first uses Singular Value Decomposition (SVD) to infer the hidden relationship between any two terms for all collected blog posts. To find the relevant terms, LSI then uses a dimensionality reduction technique to remove the noise from the result of SVD. However, LSI cannot effectively deal with the problem of polysemy because of SVD is a one-by-one mapping from a certain term to a particular document [13, 37]. PLSI uses the Expectation Maximization (EM) algorithm to estimate the most likely meanings for each term. PLSI can effectively deal with the problems of synonymy and polysemy because of the EM algorithm is a statistical inference technique that can estimate multiple parameters simultaneously [17, 57].

However, the running time for PLSI is always huge because of the EM algorithm is a time-consuming algorithm [4, 25, 40]. This of course will result in PLSI is very difficult to be applied to solve the problems of large web documents. In this paper, we propose a novel intelligent mechanism called EM with Cost-Effective Solution (EM-CES) to effectively reduce the problem of long running time in PLSI. The detail of EM-CES is shown in the following section. Overall, we can guarantee that the result obtained from PLSI with EM-CES is a cost-effective solution.

The rest of this paper is organized as follows: Section 2 presents the related work. This is followed by an overview of the method in Section 3. Section 4 shows the results with some analysis, and finally Section 5 concludes the paper and provides possible future directions.

2 Related Work

In this section, we briefly present some of the research literature related to the study of blog search, the applications of LSI and PLSI, and the termination criteria for EM algorithm.

2.1 The Study of Blog Search

Blog is an abbreviated version of weblog, which is a term used to describe websites that maintain an ongoing chronicle of information. A blog is a frequently updated personal online journal or diary kept by a blogger. Blogs range from being some of the most influential websites on the Internet like the Huffington Post, which is an online newspaper that is written by a team of professional journalists, that is worth millions, and is influential in business and politics; to niche related website like Blog Basics, which is focused exclusively on blogging related information.

Many blogs focus on a particular topic or subject area, such as politics, travel, sports, or 3C technology. Some are more generic, presenting links to all types of other sites. And others are more like personal journals, presenting the author's daily life and thoughts. Among the many sources and types of blog posts, how to effectively search these massive blog posts become a key problem in the blog study. Fortunately, there are many blog search engines available on the web such as Google Blog Search, Amatomu, Bloglines, BlogScope, IceRocket, Munax, Regator, and Technorati to help users find what users need.

In recent years, there are many researches which focus on how to effectively search the blog posts. Takama et al. [59] used a concept of keyword map to design an interactive blog search. They designed an algorithm to help the users to find some interesting blog topics, in which the users are interested on the keyword map. Kim et al. [33] proposed a trackback-rank algorithm for the effective blog search. The algorithm ranks all blog posts by calculating the reputation scores of bloggers, trackback scores, and comment scores based on the features of the blog posts. Zhu et al. [69] first analyzed the search results of a sequence of temporally-ordered queries returned by a blog search engine, and then built and maintained some blog profiles for those blogs whose posts frequently appear in the top-ranked search results. Jeong and Oh [29] proposed a blog search framework which enables a more in-depth search on a given topic by extracting the collective intelligence features in social community sites and through the query extension using these features. Kuwata et al. [31] used two processes to decide for blog posts to include some product review sentences. The first process creates a data set of certain product that contains those review sentences. Second process is a search for those review sentences. This process uses the extracted opinion tuples from one sentence of blog posts and creates the data set to decide whether a sentence containing the review sentences. Kim et al. [32] proposed an algorithm that efficiently performs a ranking for the blog posts retrieval in order to solve the uncertainty of information in blogosphere.

2.2 The Applications of LSI and PLSI

In this paper, we use LSI and PLSI to deal with the problems of synonymy and polysemy. LSI use SVD and a dimensionality reduction technique to capture the relationship between terms for all collected documents. The advantage for LSI is that it can handle the problem of identifying synonymy via the dimension-reduction technique [13]. Conversely, the disadvantage of LSI is that it cannot deal with the problem of polysemy because of SVD is only a one to one mapping from a specific term to a specific document [37]. Several researchers have successfully applied LSI to some practical applications of web mining. Ozsoy et al. [48] presented a generic extractive text summarization

system based on LSI. They applied the known and proposed LSI-based text summarization approaches to the documents with different languages. Assessing the semantic similarity of documents is an important steps in many real-world applications ranging from summarization to educational system [34, 41] to automatic detection of duplicate documents. Luh et al. [42] used LSI and the genetic algorithm to estimate Google's search engine ranking factors. In their conclusions, they suggested that multiple ranking criteria can effectively close to the ranking factors of Google search engine. Evangelopoulos [15] applied LSI to a number of applications from linguistics, psychology, cognitive science, education, information science, and the analysis of textural data. Kuo et al. [36] proposed a framework for background music recommendation based on LSI between video and music. They addressed the problem of recommending appropriate background music for a user-specified video.

PLSI uses the EM algorithm to obtain the maximum likelikhood estimates of the parameters between terms. Compared to LSI, the advantage of PLSI is that it can further handle the problem of identifying polysemy because of the EM algorithm is a statistical estimation technique that can estimate multiple parameters simultaneously [14, 17, 57]. In contrast, the running time of PLSI is very huge because of the EM algorithm is a time-consuming approach [4, 25, 40]. Many researchers have successfully applied PLSI to solve some real problems in web mining. Hennig [21] used PLSI to complete an automatic multi-document summarization task. Multi-document summarization is an increasingly important task because of the condensation of information from different sources into an informative summary helps to reduce information overload. Some researchers [66, 68] used PLSI to improve human action recognition from the query video. Human action recognition is automatically to analyze ongoing activities from unknown videos. Mesaros et al. [44] used PLSI to model the co-occurrence of overlapping sound events in audio recordings from everyday audio environments. McInerney et al. [43] used PLSI to predict the next location for a new user based on a similar pattern appears in some historical users.

2.3 The Termination Criteria for EM Algorithm

PLSI uses the EM algorithm to estimate the probability values which measure the relationships between the hidden latent factors and the two sets of objects. The EM algorithm is an iterative procedure for finding the maximum likelihood estmation's of the parameters in situations where the model depends on some missing or latent variables so that computing the maximum likelikhoog estimation is not straightforward. Each iteration of EM algorithm consists of two steps, namely the Expectation (E) step and the Maximization (M) step. In the E-step, it computes the expectation of the log-likelihood by replacing the unobservables with their conditional expectations given the current estimates of the parameters and the data; in the M-step, it maximizes the expected log-likelihood calculated in the E-step.

Although the EM algorithm can converge to a local optimization solution, it may take a very long time to reach the soution. Many researchers alternatively use different criteria to determine the algorithm should be terminated or continued. The termination criteria of the algorithm can broadly be divided into two main situations: (1) using a fixed number of iterations as the maximum predefined number of iteraionts in the algorithm [12, 19, 49, 60, 63]; (2) setting a predefined threshold to decide whether the algorithm should be terminated.

Let us now discuss the second situation as follows. Bertin et al. [2] first used the manual method to evalutate the difference performance curves generated by the algorithm. Then, a predefined threshold is decided by the average score throughout all the curves. Park and Ramamohanarao [51] introduced an efficient method to store all relevant information of the algorithm. They decided the best threshold for one group is ten to twenty percent. According to this predefined threshold, they claimed that the algorithm can save a lot of storage space. Chou and Chen [10] proposed an incremental EM model to solve the problem of online event analysis. They decided a new document should be created a new event group if its similarity with all the event group in a certain number of documents is below to a predefined threshold; otherwise, it is assigned to the event group that is the most similar. Xue et al. [67] proposed a new cross-domain text classification method to integrate the labelled and unlabelled data that are come from different but related documents into a topic-bridged EM model. They used a predefined document frequency threshold to cut down the number of features to speed up the classification process. Balog et al. [1] first used the algorithm to identify a personal name in a set of documents. Then, they applied a naive Bayes classifier to measure the similarity between a document name and a personal name. Finally, they decided that a document should be assigned to the most likely cluster of personal name if the similarity score is higher than a predefined threshold; otherwise, the document should be assigned to a new cluster. Some of other researchers used the log-likelihood function [39, 50, 62] as the performance measure. The iterations of the algorithm continue to process until the change of performance measure between two consecutive iterations is smaller than a predefined threshold.

Regardless of whether using a fixed number of iterations or a predefined threshold as a termination criterion for EM algorithm, it may causes two potential problems. First, a small number of iterations may result

in a large difference between the final and local optimization solutions. Second, a large number of iterations may result in a small improvement in these two solutions.

Some researchers [9, 35] provided a probability model for applying genetic programming to minimize the total number of individuals that need to be processed until a predefined maximum number of iterations are reached. He pointed out that there is a point after which the cost of extending a given iteration exceeds the performance obtained from the increase in the cumulative probability of success. To prevent these two potential problems as mentioned above, we use their suggestion to develop our dynamic termination strategy for EM algorithm.

3 An Overview of Our Study

In this section, we first introduce a flow chart of our study as shown in Figure 1, then explain the data source of our study. Next, we discuss some preprocessing steps to transform the data source into the input of different semantic models. After that, we describe the semantic models, namely LSI and PLSI, used in our study. Finally, we present a novel termination mechanisam, PLSI with EM-CES, to achieve a cost-effective solution.

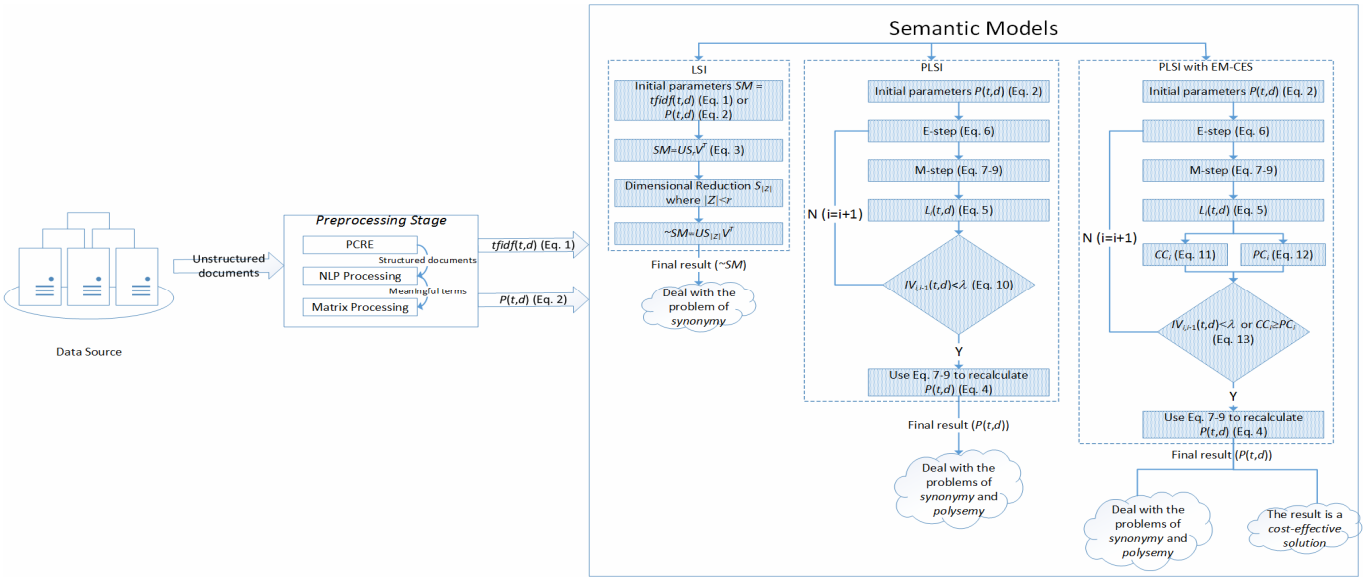


Figure 1. The flow chart of our study

The start point of our study is to fetch the relevant documents from the data source. In this paper, we take Google Blog Search's result as our data source. Google Blog Search is a specialized search engine for blogs that currently uses the search tools in Google News to search all blog posts that have been indexed by Google Blog Search. We choose Google Blog Search as our data source because of it is the most widely used and accurate blog search engine in the world [64].

Today, Google has quitely disabled the Google Blog Search home page at google.com/blogsearch and redirects it to the Google home page [55]. Now, if we want to filter content based on blog posts, we can do so by going to Google News, clicking on the search tools and selecting the "All news" drop down and checking off just "Blogs".

To significantly reduce documents' fetching time, we develop a novel web crawling agent with multi-threading capability. In short, we can crawl the documents as much as possible in parallel. Theoretically, when the factors of network latency and document processing time are not considered, the fetching time for many documents is the same as for a single document.

3.1 The Preprocessing Stage

In the preprocessing stage, we totally use the Perl Compatible Recular Expression (PCRE), Natural Language Processing (NLP), and Matrix Processing steps to transform the data source into the input of different semantic models.

First of all, we must establish a method to convert unstructured HTML documents into structured documents because of the document returned from the data source is an unstructured HTML document [8]. In this paper, we use PCRE [20], which is a regular expression pattern matching library, to do the conversion process. Each structured document for our study consists of a document title, snippet, and URL.

Secondly, we use some NLP techniques to transform each sentence in the structured document into a series of meaningful terms. We define that a term is a meaningful term if and only if each term satisfies the following three conditions: (1) it does not contain any stop words; (2) it only contains the root word; (3) it does not contain any non-word tokens. In this study, we use the following NLP techniques to satisfy the above three conditions: Stop-Word, Stemming, and

Non-Word. We first use 421 stop words such as articles, prepositions, pronouns, and so on, as suggested by Fox [18], to filter out unimportant words. We also use Porter's stemming algorithm [53] to obtain the root word for a given word. Finally, all non-word characters, such as puncuation and whitespace with nothing, are stripped off.

Lastly, in the Matrix step, we transform all meaningful terms into the structured matrix data, which is the input of LSI and PLSI semantic models. LSI and PLSI are the well known matrix models [22]; that is, we need to transform the meaningful terms into a matrix form. In this step, we use the Term Frequency-Inverse Document Frequency (TFIDF) and Probability (P) methods to do the transformation task.

The TFIDF method, as shown in the following equation, gives a high weight to the meaningful terms occurring frequently in the document but rarely in the rest of the corpus.

$$tfidf(t,d) = tf(t,d) \times \log(|D| / |\{d \in D : t \in d\}|) \quad \textbf{(1)}$$

where $tfidf(t,d)$ represents the TFIDF weight of term t in document d; $tf(t,d)$ is the number of times that term i occurs in document j; $|D|$ is the total number of documents in the corpus D; $|\{d \in D : t \in d\}|$ is the number of documents where term t appears. The P method sets a probability weight of term t in document d as shown in the following equation.

$$P(t,d) = tfidf(t,d) / \sum_k tfidf(k,d) \quad \textbf{(2)}$$

where $P(t,d)$ represents the probability weight of term t in document d and the denominator of $P(t,d)$ is the sum of each tfidf's weight in the document d.

3.2 The Semantic Models Stage

In the semantic models stage, we use the LSI and PLSI semantic models to deal with the problems of synonymy and polysemy. LSI first uses SVD to decompose the structured matrix data SM into the product of three other matrices, as shown in the following equation.

$$SM = USV^T \quad \textbf{(3)}$$

where $U = (u_1, u_2, \ldots, u_r)$ is a $|T| \times r$ matrix of left singular vectors; $S = \text{diag}(s_1, s_2, \ldots, s_r)$ is a $r \times r$ diagonal matrix; $V = (v_1, v_2, \ldots, v_r)$ is a $|D| \times r$ matrix of right singular vectors. LSI then finds a low-rank reduction matrix $\sim SM$, which is smaller and less noisy than SM to filter noisy data and absorb synonymy. For a fixed $|Z| < r$, using a truncated SVD technique that conserves $|Z|$ largest singular values in S matrix and set others to be zero to approximate original SM matrix.

PLSI uses an aspect model to identify the hidden semantic relationship among terms and documents [23]. It first calculates the join probability of an observed pair, $P(t,d)$, by summing overall all possible choices of z from which the observation has been generated, as shown in the following equation; where $p(t|z)$ denotes the posterior probability of a particular term t given the observation laten topic z, $p(z)$ denotes the probability that z has observed, $p(d|z)$ denotes the posterior probability of a particular document d given z.

$$P(t,d) = \sum_Z p(t|z)p(z)p(d|z) \quad \textbf{(4)}$$

To obtain the final result of $P(t,d)$, PLSI follows the likilihood principle to estimate the parameters of $p(t|z)$, $p(z)$, and $p(d|z)$ by maximum the likihood function's value at iteration i, $L_i(t,d)$, as shown in the following equation.

$$L_i(t,d) = \sum_t \sum_d SM \times \log(p(t,d)) \quad \textbf{(5)}$$

The standard procedure for maximum likelihood estimation in PLSI is the EM algorithm. Generally, two steps are needed to perform in the algorithm alternately:

- E-step, where the posterior probability of z is calculated based on the current estimates of conditional probability, as shown in the following equation.

$$p(z|t,d) = p(t|z)p(z)p(d|z) / \sum_z p(t|z)p(z)p(d|z) \quad \textbf{(6)}$$

- M-step, where the estimated conditional probabilities are updated and used to maximize the total likelihood function based on the posterior probability calculated in E-step, as shown in the following equations.

$$p(t|z) = \sum_d SM \times p(z|t,d) / \sum_t \sum_d SM \times p(z|t,d) \quad \textbf{(7)}$$

$$p(z) = \sum_t \sum_d SM \times p(z|t,d) / \sum_t \sum_d SM \quad \textbf{(8)}$$

$$p(d|z) = \sum_t SM \times p(z|t,d) / \sum_t \sum_d SM \times p(z|t,d) \quad \textbf{(9)}$$

3.3 The Detail of PLSI with EM-CES

The computational time of PLSI is higly dependent on the EM algorithm. Hofmann et al. [24] has been proven that the time complexity of PLSI is $O(|T| \times |D| \times |Z|)$, where $O(|T| \times |D|)$ is the time complexity of EM algorithm for each iteration. However, in the current Internet age, the total number of terms ($|T|$) and documents ($|D|$) are both very huge. At the same time, the total number of laten topics ($|Z|$) follows the growth of $|T|$ and $|D|$ [27]. When PLSI is applied to solve a large-scale information retrieval problem, in such huge $|T|$, $|D|$, and $|Z|$, PLSI is prone to the performance degradation problem.

According to the literature review, the termination criteria for EM algorithm are divided into two situations: (1) it converges to a local optimization solution, and (2) it reaches the maximum allowable number of iterations. Next, let use discuss these two situations as follows.

Situation 1: it converges to a local optimization solution. In this situation, we define that the local

optimal solution is reached, as shown in the following equation, if the improvement value between two consecutive iterations is less than a predefined small threshold value.

$$IV_{i,i-1}(t,d) \le \lambda \text{ where } IV_{i,i-1}(t,d) = L_i(t,d) - L_{i-1}(t,d) \quad \textbf{(10)}$$

where $L_i(t,d)$, as mentioned in equation (5), is the the likelihood function's value at iteration i; λ is the predefined threshold value; $IV_{i,i-1}(t,d)$ is the improvement value between iterations i and i-1.

Situation 2: it reaches the maximum allowable number of iterations. If we set a small number of iterations to perform the EM algorithm, it may result in a large difference between the final and local optimal solutions. Conversely, a large number of iterations to perform the algorithm may result in a small improvement in these two solutions. The key point of these two problems is that cost and performance are always on the opposite sides.

In our study, we use the suggestion by Koza [35] that there is a point after which the cost of extending a given iteration exceeds the performance to develop our EM-CES termination criterion. According to there suggestion, we need to define two important curves, one is the cost curve and the other is the performance curve.

The cost curve is defined as the number of successive iterations for which the performance is not improved. According to this definition, the cost curve for iteration i, CC_i, is shown in the following equation, where the right-hand side of the "if" condition in equation (11) is the average value of all improvement values for each iteration.

$$\begin{aligned} CC_i &= CC_{i-1} + 1, \\ \text{if } IV_{i,i-1}(t,d) &\le avg(\textstyle\sum_{\gamma \le i} IV_{\gamma,\gamma-1}(t,d)) \end{aligned} \quad \textbf{(11)}$$

The performance curve for iteration i, PC_i, is defined as following, where $|Z|$ is the total number of laten topics, $\sigma(IV_{i,i-1}(t,d))$ is the standard deviation of all $IV_{\gamma,\gamma-1}(t,d)$ values, $avg(\Sigma_{\gamma \le i}\, \sigma(IV_{\gamma,\gamma-1}(t,d)))$ is the average value of all $\sigma(IV_{i,i-1}(t,d))$ values for each iteration.

$$PC_i = \left[\frac{IV_{i,i-1}(t,d)}{avg(\sum_{\gamma \le i} IV_{\gamma,\gamma-1}(t,d))} \times |z| \times \frac{\sigma(IV_{i,i-1}(t,d))}{avg(\sum_{\gamma \le i} \sigma_{\gamma,\gamma-1}(t,d))} \right] \quad \textbf{(12)}$$

We then disscuss the performance curve as follows. According to the above discussion, the time complexity of PLSI is $O(|T| \times |D| \times |Z|)$. Obviously, the total number of iterations required by PLSI is closely related to the total number of laten topics $|Z|$; thus, we set $|Z|$ as a dominant parameter of the performance curve.

We also use the rate of historical performance improvement ($IV_{i,i-1}(t,d)/avg(\Sigma_{\gamma \le i} IV_{\gamma,\gamma-1}(t,d))$) and the rate of variable performance improvement ($\sigma(IV_{i,i-1}(t,d))/avg(\Sigma_{\gamma \le i}\, \sigma(IV_{\gamma,\gamma-1}(t,d)))$) as the other two domiant parameters of the performance curve. We assume that there is an insignificant performance improvement or variable performance improvement in the likelihood function if either the rate of historical performance improveent or the rate of variable performance improvement is less. The situation 2 can use Figure 2 to represent it.

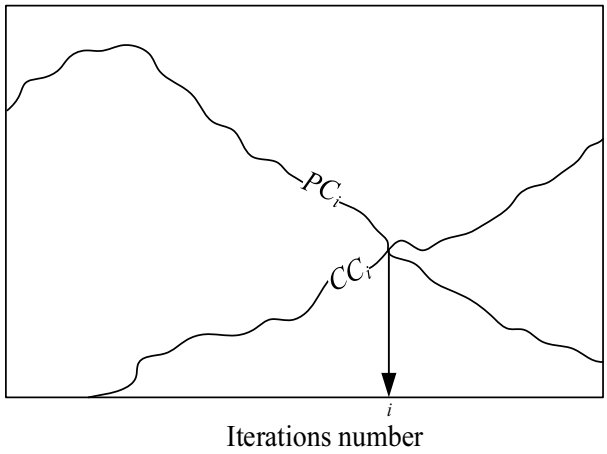


Figure 2. A diagram illustration of the cost and performance curves

Lastly, by the definition of the above two situations, we can define our dynamic termination strategy for EM algorithm as follows.

$$IV_{i,i-1}(t,d) \le \lambda \text{ or } CC_i \ge PC_i \quad \textbf{(13)}$$

4 The Results and Discussion of the Experiment

In this section, we perform two experiments to illustrate the effectiveness and efficiency of the proposed method. In the first experiment, we want to compare the performance of different semantic models and WordNet. In the second experiment, we perform a simulation to verify that the result obtained from our PLSI with EM-CES mechanisam is a cost-effective solution.

4.1 Experimental Data and Measures

In this study, the experimental data is selected from top 10 searched keywords on Google and Yahoo in 2012, 2013, and 2014. By screening all keywords, we delete some repeating keywords and retain only one keyword. The full set of experimental data is shown in [5]. We first send the selected keywords to Google Blog Search and then do all tasks in Figure 1. Actually, for each keyword, Google Blog Search can return from 350 to 400 search results; thus, we respectively choose 10, 20, 40, 80, 160, 240, and 350 documents as the data source of our experiment.

Due to the result of LSI and PLSI is a matrix form, we need to find some similarity functions to calculate the similarity measure between two term vectors in a

term-by-document matrix. The similarity function can be any similarity measure. The most commonly used are cosine and correlation coefficient similarity measures [6]; thus, in this paper, we use these two similarity measures to evaluate the performance of different experiments.

Next, we use an example to illustrate how to use the similarity functions to calculate the average similarity measure of the matrix. Assume Table 1 is the final result of the semantic model when the user's query is processed.

Table 1. The final result of the Y semantic model, where $Y \in$ {LSI, PLSI}

	d_1	d_2	d_3
t_1	0.56	0.12	0.33
t_2	0.34	0.25	0.25
t_3	0.74	0.33	0.15

We first calculate the cosine similarity score of the term vectors t_1 (<0.56, 0.12, 0.33>) and t_2 (<0.34, 0.25, 0.25>). The calculation process is shown below.

$$Cos(t_1,t_2) = \frac{.56\times.34 + .12\times.25 + .33\times.25}{\sqrt{.56^2+.12^2+.33^2}\times\sqrt{.34^2+.25^2+.25^2}} = 0.93424$$

Similarly, we also can calculate the cosine similarity scores for $Cos(t_1,t_3)$=0.92443 and $Cos(t_2,t_3)$=0.91938. We now can calculate the average cosine similarity measure of the matrix by an average of these three cosine similarity scores, so the average cosine similarity measure of the matrix is 0.92602.

We then calculate the correlation coefficient similarity score of the term vectors t_1 and t_2 where the average values of t_1 and t_2 are $\bar{t}_1$ =0.34 and $\bar{t}_2$ =0.28, respectively. The calculation process is shown below.

$$\rho(t_1,t_2) = \frac{(.56-.34)\times(.34-.28)+(.12-.34)\times(.25-.28)+(.33-.34)\times(.25-.28)}{\sqrt{[(.56-.34)^2+(.12-.34)^2+(.33-.34)^2]\times[(.34-.28)^2+(.25-.28)^2+(.25-.28)^2]}} = 0.87884$$

Similarly, we also can calculate the correlation coefficient similarity scores for $\rho(t_1,t_3)$=0.69701 and $\rho(t_2,t_3)$=0.95468. We now can calculate the average correlation coefficient similarity measure of the matrix by an average of these three correlation coefficient similarity scores, so the average correlation coefficient similarity measure of the matrix is 0.84351.

4.2 Experiment with Different Semantic Models and WordNet

We input a keyword "mobile phone jb" to Google Blog Search to carefully detail and explain the synonymous and polysemous problems how to impact the performance on different semantic models. Table 2 shows the results of cosine and correlation coefficient measures for different analysis models. Let us look at the table; we observe that the cosine and correlation coefficient measures for LSI and PLSI semantic models are significantly better than non-semantic TFIDF model. Next, we analyze the reason for it.

Table 2. The results of cosine and correlation coefficient measures for different analysis models

(a) TFIDF matrix					
	d_1	d_2	d_3	d_4	d_5
mobile phone	0	0	0	0	0
jb	0	0.67	0	0.44	0.66
jailbreaking	0.67	0	0.67	0	0.44
cellular phone	0	0.48	0.19	0.10	0.29
jelly been	0	0.80	0	1.59	0
Cosine-0.48257, Correlation Coefficient-0.40253					
(b) LSI matrix					
	d_1	d_2	d_3	d_4	d_5
mobile phone	0.53	0.47	0.28	0.38	0.61
jb	0.25	0.31	0.42	0.36	0.19
jailbreaking	0.28	0.52	0.33	0.28	0.22
cellular phone	0.46	0.43	0.24	0.37	0.57
jelly been	0.32	0.37	0.52	0.54	0.25
Cosine-0.91671, Correlation Coefficient-0.66927					
(C) PLSI matrix					
	d_1	d_2	d_3	d_4	d_5
mobile phone	0.38	0.22	0.35	0.37	0.27
jb	0.32	0.35	0.27	0.29	0.31
jailbreaking	0.45	0.37	0.40	0.43	0.34
cellular phone	0.43	0.36	0.44	0.40	0.35
jelly been	0.38	0.29	0.37	0.41	0.34
Cosine-0.98961, Correlation Coefficient-0.70190					

Let us look at the table again in detail, we find that two terms, "mobile phone" and "cellular phone" have the feature of synonymy. Similarly, the terms, "jb" and "jailbreaking" (or "jelly bean"), also have the feature of synonymy; that is, the abbreviation of "jailbreaking" or "jelly bean" is "jb". In this experiment, the cosine and correlation coefficient measures for LSI are significantly improved to 0.91671 and 0.66927, respectively, compared to TFIDF model, respectively. That is, by filtering the noise data from the non-semantic TFIDF matrix, LSI can group the semantic-related terms with the same topic together to enhance the retrieval performance.

Moreover, the term "jb" is also a polysemous term because of it has at least two distinct meanings, "jailbreaking" and "jelly bean". Looking the result of PLSA, we find that the cosine and correlation coefficient measures for PLSI are further enhanced to 0.98961 and 0.70190, respectively, compared to LSI model. This performance improvement is because PLSI can further deal with the problem of polysemy. This result also echoed the finding of Ishida and Ohta [28]; that is, LSI and PLSI can effectively deal with the problem of synonymy, but LSI lacks capability to deal with the problem of polysemy, because, by using SVD technology, a row vector in a matrix can only represent

a term. PLSA uses the aspect model to estimate the joint probability of terms and documents. By using the latent probability method, we can clearly distinguish different meanings and types between terms so PLSA can gracefully deal with the problem of polysemy.

Next, we conduct an extended experiment on a large-scale dataset. For the above-mentioned 49 keywords, we also use Google Blog Search to crawl the HTML documents, and then perform the preprocessing stage in Figure 1 to produce a term-by-document matrix required by different analysis models. We also use the cosine and correlation coefficient measures to measure the performance of different analysis models.

The performance of different analysis models based on different number of latent topics as shown in the subfigures (a) and (b) of Figure 3. Each dot in these subfigures is the average value of all different number of documents (10, 20, 40, 80, 160, 240, and 350). According to the results of these two subfigures, the similarity value is decreased along with the total number of latent topics $|Z|$ is increased. The best and worst semantic models are P-PLSI and TFIDF-LSI, respectively. Note that, in Figure 3, *X*-*Y* denotes that the input of *Y* semantic model is the result of *X* method where $X \in$ {TFIDF, P}, $Y \in$ {LSI, PLSI}, and TFIDF not *X*-*Y* denotes the non-semantic model's result.

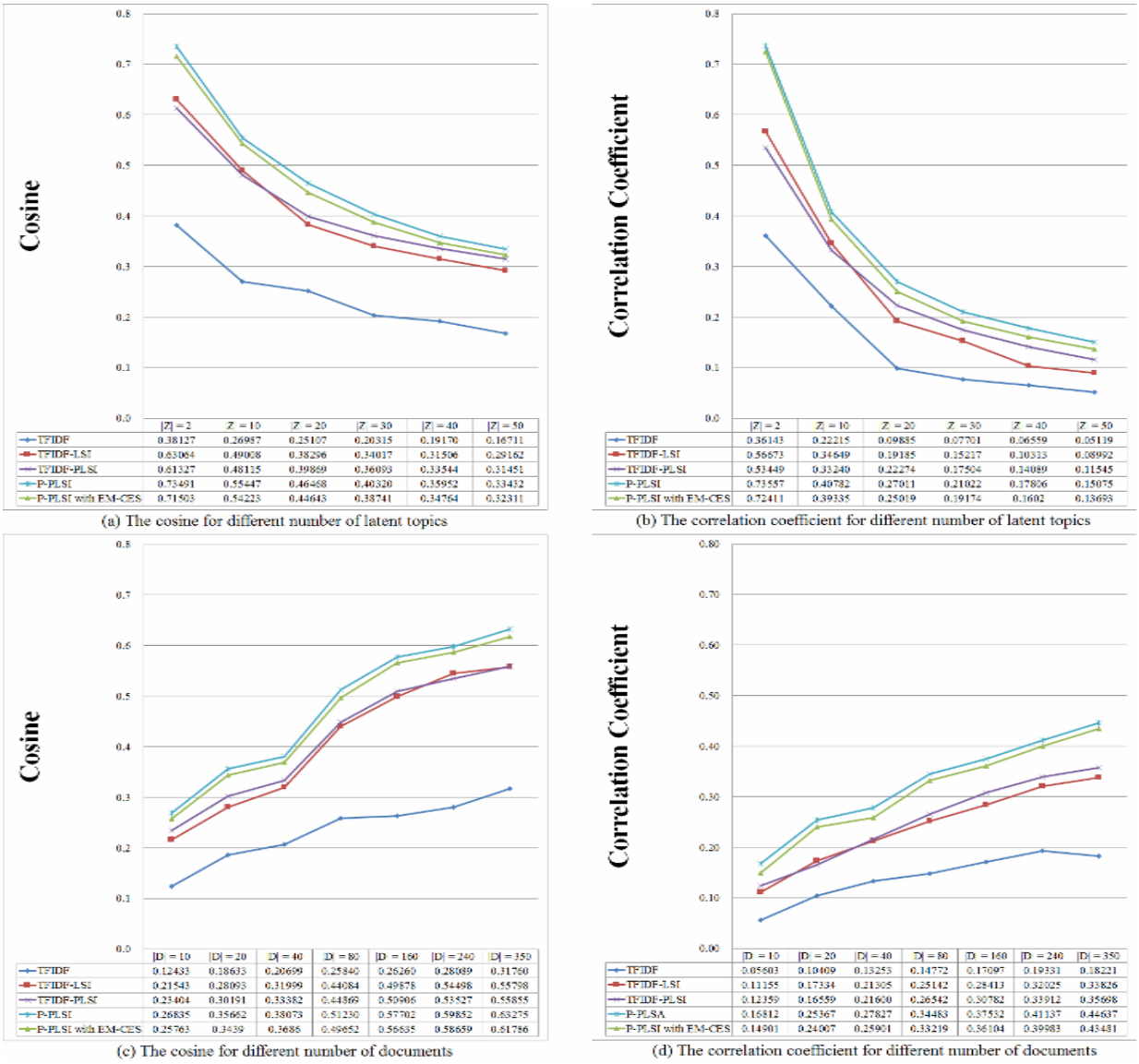


Figure 3. The performance comparison for different analysis models

The similarity difference between TFIDF-LSI and P-PLSI is largest when $|Z|$ is equal to 2, and the difference is getting smaller when $|Z|$ is getting larger. For some special cases ($|Z| = 2$ or $|Z| = 10$), the performance of TFIDF-LSI is better than TFIDF-PLSI, but in other cases ($|Z|>10$), the performance of TFIDF-PLSI is better than TFIDF-LSI. This implies that PLSI has a better performance when $|Z|$ is getting larger. Compared to P-PLSI and TFIDF-PLSI, we find that the performance of P-PLSI is always better than TFIDF-PLSI and this implies that a mixed input method P, which is a probability processing of TFIDF as described in section 3.1, can be used to improve the performance.

Similarly, the performance of different analysis models based on different number of documents as shown in the subfigures (c) and (d) of Figure 3. Each dot in theses subfigures is the average value of all different number of topics (2, 10, 20, 30, 40, and 50). According to the results of these two subfigures, the performance is increased along with the number of documents $|D|$ is increased. This implies that the performance is getting better when the number of document is getting larger. According to the above analysis, the performance derived from the PLSI-like model is better than the LSI-like model, and this implies that PLSI can achieve a better semantic processing ability than LSI.

Let us now compare the models of P-PLSI and P-PLSI with EM-CES, the performance difference is between 1.067% to 1.992%. However, according to the result of experiment on the cost-effective solution as shown in the next experiment, P-PLSI needs more execution time than P-PLSI with EM-CES at least 240%. This implies that the performance improvement of PLSI is negligible after iteration i decide by PLSI with EM-CES. PLSI with EM-CES can significantly reduce the execution time and maintain the performance. That is, we can guarantee that the result generated from our PLSI with EM-CES model is a cost-effective solution.

Finally, let use compare the result of non-semantic TFIDF and semantic X-Y models. According to the result of Figure 3, we find that the performance of X-Y is improved between 3.754% to 26.409% compared to TFIDF model. This implies that, by using the semantic analysis method, we can effectively deal with the problems of synonymy and polysemy.

WordNet is a lexical database of English which groups lexical units, nouns, verbs, adjectives and adverbs, according to their semantic and lexical relations. In the latest version, it contains 155287 words organized in 117,659 synsets (a set of words having the same or close meaning) for a total of 206941 word-sense pairs [65]. One of the traditional ways to solve the problems of synonymy and polysemy is to use WordNet as a corpus-based dictionary to measure the similarity of two words [16, 30, 56]. Next, we use WordNet as a word corpus to measure the average similarity of the matrix.

The detailed process of WordNet processing is described as follows: we first use the preprocessing stage of Figure 1 to generate a TFIDF matrix based on the user's query. Then, for any terms t_x and t_y in a document, we update the weights of t_x and t_y to the maximum weight for the weights of t_x and t_y if the weights of t_x and t_y are not 0 where t_x and t_y are the same synset in the WordNet corpus. To generate a TFIDF-WordNet, we repeatedly update the weights for each cell of the original TFIDF matrix until all cells are compared. Table 3 shows the results of cosine and correlation coefficient measure for TFIDF, TFIDF-WordNet, and TFIDF-LSA.

Table 3. The comparison of TFIDF, TFIDF-WordNet, and TFIDF-LSA

Cosine						
	$\|Z\|$=2	$\|Z\|$=10	$\|Z\|$=20	$\|Z\|$=30	$\|Z\|$=40	$\|Z\|$=50
TFIDF	0.38127	0.26987	0.25107	0.20315	0.19170	0.16711
TFIDE-WordNet	0.41265	0.31234	0.28124	0.23141	0.22178	0.19144
TFIDF-LSI	0.63064	0.49008	0.38296	0.34017	0.31506	0.29162
Improved Rate	52.827%	56.906%	36.168%	46.999%	42.060%	52.330%
Correlation Coefficient						
	$\|Z\|$=2	$\|Z\|$=10	$\|Z\|$=20	$\|Z\|$=30	$\|Z\|$=40	$\|Z\|$=50
TFIDF	0.36143	0.22215	0.09885	0.07701	0.06559	0.05119
TFIDE-WordNet	0.38245	0.24135	0.12351	0.09877	0.08912	0.07643
TFIDF-LSI	0.56673	0.34649	0.19185	0.15217	0.10313	0.08992
Improved Rate	48.184%	43.563%	55.332%	54.065%	15.720%	17.650%

In the table, the row of "Improved Rate" represents the improvement rate of the LSI semantic model (TFIDF-LSI) compared to the WordNet dictionary model (TFIDF-WordNet). For example, in the cosine measure, the improvement rate of TFIDF-LSI is (0.63064-0.41265)/0.41265=52.827% for $|Z|$=2 compared to TFIDF-WordNet. Totally, the improvement rate of the LSI semantic model can range from 15.720% to 56.906%. The reasons are as following: (1) on the one hand, in the WordNet corpus, it only covers a limited amount of words, but in the field of blog search, they have many new terminologies not in the WordNet corpus; (2) on the other hand, in the WordNet corpus, it is based on the word rather than the term as the basic element of NLP processing, however the meaning of word is always less than the whole term. This implies that the performance of the semantic model is significantly better than the dictionary model in the field of blog search.

4.3. Experiment on the Cost-effective Solution

In this experiment, we want to verify the result generated from our PLSI with EM-CES model is a cost-effective solution. We randomly generate the following parameters: $|T|$ (total number of terms), $|D|$ (total number of documents), $|Z|$ (total number of latent topics), and the elements in TFIDF matrix. Interested readers can simulate this experiment at http://cayley.sytes.net/simulation/em-ces.php. All other parameters, including i, $IV_{i,i-1}(t,d)$, $avg(\Sigma_{\gamma \le i} IV_{\gamma,\gamma-1}(t,d))$, $\sigma(IV_{i,i-1}(t,d))$, $avg(\Sigma_{\gamma \le i}\sigma(IV_{\gamma,\gamma-1}(t,d)))$, PC_i, CC_i, and $L_i(t,d)$, are defined in section 3.

Figure 4 is a simulation run for our PLSI with EM-CES model. In the first situation, $IV_{i,i-1}(t,d) \le \lambda$, we set λ equal to 0.01, and it means that the situtation is met if the improvement value between two consecutive iterations is less than or equal to a relatively small value λ. In Figure 4, the local optimal solution is reached at iteration iteration 71 and its likelihood function's value is 821.461 ($L_{71}(t,d)$ = 821.461). In the second situtation, $CC_i \ge PC_i$, our PLSI with EM-CES

model stops to run at iteration 30 and its likelihood function's value is 812.247 ($L_{30}(t,d)$ = 812.247). In this simulation, PLSI with EM-CES decides to stop the EM algorithm at iteration 30 rather than 71 because of it takes an extra 41 (71-30) iterations to result in a slightly 9.214 improvement. This implies that the performance improvement is negligible after iteration i decided by PLSI with EM-CES.

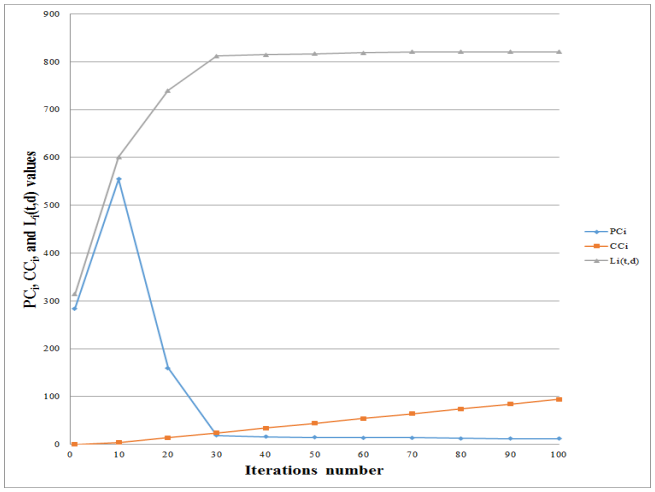


Figure 4. A simulation run for our PLSI with EM-CES model

To verify the solution derived from the second situtation is a cost-effective solution, we use a Cost-Benefit (CB) metric to compare some predefined number of iterations and the iteration i decided by the second situtation ($i = i$). In this experiment, we perform 1000 simulation runs and use the following predefined number of iterations: i=20, i=40, i=60, i=80, and i=100.

In the CB metric, we need to define two submetrics, Cost (C) and Benefit (B). The formula of C submetric is shown below, where $R_i(t,d)$ and $R_{optimal}(t,d)$ denote the required number of iterations calculated by iteration i and the local optimal solution, respectively.

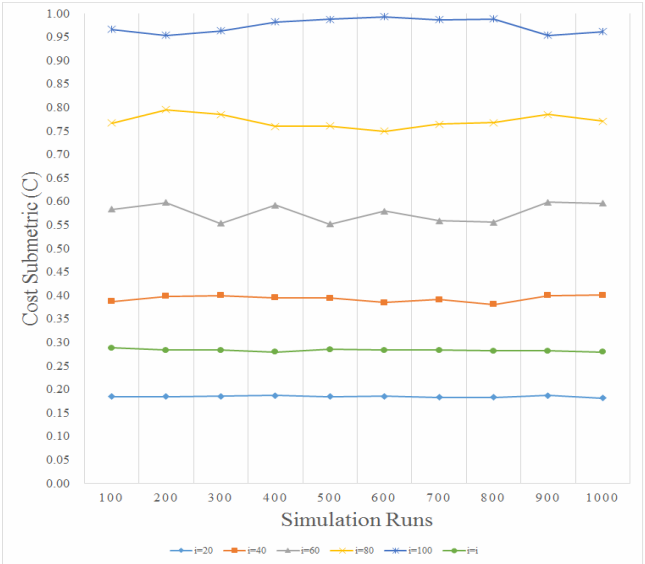
$$C_i = R_i(t,d) / R_{optimal}(t,d) \quad \textbf{(14)}$$

The subfigure (a) of Figure 5 is the distribution over C submetric. To understand and simplify the complexity of Figure 5, each dot in the figure represents the average value of 100 simulation runs. The average C values for different number of iterations are 0.18543 (i=20), 0.39380 (i=40), 0.57742 (i=60), 0.77120 (i=80), 0.97433 (i=100), and 0.28369 (i=i), respectively.

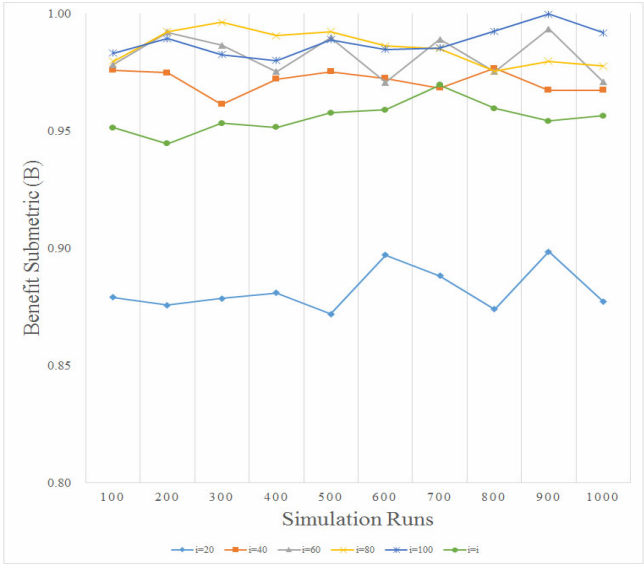
The formula of B submetric is shown below, where $L_i(t,d)$ and $L_{optimal}(t,d)$ denote the likelihood function's value calculated by iteration i and the local optimal solution, respectively.

$$B_i = L_i(t,d) / L_{optimal}(t,d) \quad \textbf{(15)}$$

The subfigure (b) of Figure 5 is the distribution over B submetric. The average B values for different number of iterations are 0.88224 (i=20), 0.97126 (i=40), 0.98223 (i=60), 0.98562 (i=80), 0.98793 (i=100), and 0.95582 (i=i), respectively.



(a) The distribution over C submetric



(a) The distribution over B submetric

Figure 5. The distribution over C and B submetrics

Although a lower C value will result in using less computer resources, it will result in a lower B value. Let us now find the relationship between C and B. For any pair of different number of iterations i_1 and i_2, where $i_1 \geq i_2$, we define our CB metric as follows.

$$CB_{i_1,i_2} = (B_{i_1} - B_{i_2}) / (C_{i_1} - C_{i_2}) \quad \textbf{(16)}$$

We use i_2 = 20 as our performance benchmark to verify the solution derived from the second situtation is a cost-effective solution. Table 4 shows the different $CB_{i1,20}$ values based on i_2 = 20. According to the result of Table 4, we find that the CB value derived from the second situtation, $i_1 = i$, is always significantly better than other i_1 values at least 32.161% (0.74883 - 0.42722). Moreover, the CB value declines very significantly when the required number of iterations is larger than $i_1 = i$. Therefore, the solution derived from our PLSI with EM-CES is a cost-effective solution.

Table 4. The different CB values based on i_2 = 20

	i_2=20	i_1=40	i_1=60	i_1=80	i_1=100	i_1=i
B	0.88224	0.97126	0.98223	0.98562	0.98793	0.95582
C	0.18543	0.39380	0.57742	0.77120	0.97433	0.28369
CB		0.42722	0.25508	0.17649	0.13397	0.74883

5 Conclusions and Future Work

In the blog search study, it often faces two semantic problems, synonymy and polysemy. In this paper, we applied LSI and PLSI to deal with these two semantic problems. LSI can identify the synonymous relationship between terms. PLSI can further deal with the polysemous problem. Although, PLSI can gracefully deal with these two semantic problems, it needs a huge computing time. To solve the problem of computing time, in this paper, we proposed a novel termination mechanism for PLSI, called PLSI with EM-CES, to dynamically determine the required number of iterations for PLSI. According to the analysis and experiment results, the final result derived from PLSI with EM-CES can not only deal with the problems of synonymy and polysemy but also reach a cost-effective solution.

Due to EM-CES is a post-processing mechanism for EM algorithm, so that, in theory, it can be applied to solve any EM-like problems, such as data clustering in data learning and machine learning [38], statistical natural language processing [7], probabilistic context-free grammars [68], and hidden Markov model estimation [26]. Furthermore, we intend to apply EM-CES to solve other EM-like problems to drastically reduce the computing time.

Acknowledgements

We would like to thank anonymous reviewers of the paper for their constructive comments which help us to improve the paper in serveral ways. This work was supported in part by Ministry of Science and Technology, Taiwan under Grant Most 107-2410-H-259-016, 106-2410-H-259-011, 105-2221-E-259-030 & 104-2221-E-259-038.

References

[1] K. Balog, L. Azzopardi, M. de Rijke, Personal Name Resolution of Web People Search, *Proceedings of WWW2008 workshop NLP Challenges in the Information Explosion Era (NLPIX 2008)*, Beijing, China, 2008, pp. 1-10.

[2] N. Bertin, R. Badeau, E. Vincent, Enforcing Harmonicity and Smoothness in Bayesian Non-Negative Matrix Factorization Applied to Polyphonic Music Transcription, *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 18, No. 3, pp. 538-549, March, 2010.

[3] D. Best, Web 2.0 Next Big Thing or Next Big Internet Bubble?, http://zh.scribd.com/doc/4635236/Web-2-0#scribd.

[4] A. V. Brahmane, A. Amune, A Survey of Dynamic Distributed Network Intrusion Detection Using Online Adaboost-Based Parameterized Methods, *International Journal of Innovative Research in Advanced Engineering*, Vol. 1, No. 9, pp. 256-262, October, 2014.

[5] CayleyGroup, The Full Queries Set for Google Blog Search, http://cayley.sytes.net/list/full%20set%20queries.pdf.

[6] S. P. Charaniya, W.-S. Hu, G. Karypis, Mining Bioprocess Data: Opportunities and Challenges, *Trends in Biotechnology*, Vol. 26, No. 12, pp. 690-699, December, 2008.

[7] N. Chater, C. D. Manning, Probabilistic Models of Language Processing and Acquisition, *Trends in Cognitive Sciences*, Vol. 10, No. 7, pp. 335-344, July, 2006.

[8] L.-C. Chen, Improving the Performance of Wikipedia Based on the Entry Relationship between Articles, *Journal of Internet Technology*, Vol. 19, No. 3, pp. 711-723, May, 2018.

[9] L.-C. Chen, C.-J. Luh, C. Jou, Generating Page Clippings from Web Search Results Using a Dynamically Terminated Genetic Algorithm, *Information Systems*, Vol. 30, No. 4, pp. 299-316, June, 2005.

[10] T.-C. Chou, M. C. Chen, Using Incremental Plsi for Threshold-Resilient Online Event Analysis, *IEEE Transactions On Knowledge and Data Engineering*, Vol. 20, No. 3, pp. 289-299, March, 2008.

[11] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, R. Harshman, Indexing by Latent Semantic Analysis, *Journal of the American Society for Information Science*, Vol. 41, No. 6, pp. 391-407, September, 1990.

[12] Y. Dorfan, G. Hazan, S. Gannot, Multiple Acoustic Sources Localization Using Distributed Expectation-Maximization Algorithm, *Proceedings of the 2014 Hands-free Speech Communication and Microphone Arrays*, Villers-les-Nancy, France, 2014, pp. 72-76.

[13] S. T. Dumais, Latent Semantic Analysis, *Annual Review of Information Science and Technology*, Vol. 38, No. 1, pp. 188-230, 2004.

[14] T. Eda, M. Yoshikawa, T. Uchiyama, T. Uchiyama, The Effectiveness of Latent Semantic Analysis for Building up a Bottom-up Taxonomy from Folksonomy Tags, *World Wide Web*, Vol. 12, No. 4, pp. 421-440, December, 2009.

[15] N. E. Evangelopoulos, Latent Semantic Analysis, *Wiley Interdisciplinary Reviews: Cognitive Science*, Vol. 4, No. 6, pp. 683-692, November/December, 2013.

[16] C. Fellbaum, Wordnet: *An Electronic Lexical Database*, MIT Press, 1998.

[17] R. Fernandez-Beltran, F. Pla, Incremental Probabilistic Latent Semantic Analysis for Video Retrieval, *Image and Vision Computing*, Vol. 38, No. C, pp. 1-12, June, 2015.

[18] C. Fox, A Stop List for General Text, *ACM SIGIR Forum*, Vol. 24, No. 1-2, pp. 19-35, Fall, 1989.

[19] R. GajšEk, F. Mihelič, S. DobrišEk, Speaker State Recognition Using an Hmm-Based Feature Extraction Method, *Computer Speech & Language*, Vol. 27, No. 1, pp. 135-150, January, 2013.

[20] P. Hazel, Pcre- Perl Compatible Regular Expressions, http:// www.pcre.org/pcre.txt.

[21] L. Hennig, Topic-Based Multi-Document Summarization with Probabilistic Latent Semantic Analysis, *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, Borovets, Bulgaria, 2009, pp. 144-149.

[22] T. Hofmann, Unsupervised Learning by Probabilistic Latent Semantic Analysis, *Machine Learning*, Vol. 42, No. 1-2, pp. 177-196, Juauary, 2001.

[23] T. Hofmann, Latent Semantic Models for Collaborative Filtering, *ACM Transactions on Information Systems*, Vol. 22, No. 1, pp. 89-115, January, 2004.

[24] T. Hofmann, B. Schölkopf, A. J. Smola, Kernel Methods in Machine Learning, *The Annals of Statistics*, Vol. 36, No. 3, pp. 1171-1220, June, 2008.

[25] J.-W. Hsieh, L.-C. Chen, S.-Y. Chen, D.-Y. Chen, S. Alghyaline, H.-F. Chiang, Vehicle Color Classification under Different Lighting Conditions through Color Correction, *IEEE Sensors Journal*, Vol. 15, No. 2, pp. 971-983, February, 2015.

[26] D. Hsu, S. M. Kakade, T. Zhang, A Spectral Algorithm for Learning Hidden Markov Models, *Journal of Computer and System Sciences*, Vol. 78, No. 5, pp. 1460-1480, September, 2012.

[27] M. Inoue, The Remarkable Search Topic-Finding Task to Share Success Stories of Cross-Language Information Retrieval, *SIGIR 2006 Workshop: New Directions In Multilingual Information Access*, Seattle, USA, 2006, pp. 61-64.

[28] K. Ishida, T. Ohta, An Approach for Organizing Knowledge According to Terminology and Representing It Visually, *IEEE Transactions on Systems, Man, and Cybernetics- Part C: Applications and Reviews*, Vol. 32, No. 4, pp. 366-373, November, 2002.

[29] O.-R. Jeong, J. Oh, Social Community Based Blog Search Framework, in: H. Yu, G. Yu, W. Hsu, Y. S. Moon, R. Unland, J. Yoo (Eds.), *Database Systems for Advanced Applications, DASFAA 2012, Lecture Notes in Computer Science*, Vol. 7240, Springer, Berlin, 2012, pp. 130-141.

[30] N. K. Jha, A. Jethva, N. Parmar, A. Patil, A Review Paper on Deep Web Data Extraction Using Wordnet, *International Research Journal of Engineering and Technology*, Vol. 3, No. 3, pp. 1003-1006, March, 2016.

[31] H. Kuwata, M. Oka, H. Mori, Searching Blog Sites with Product Reviews, in: S. Yamamoto (Ed.), *Human Interface and the Management of Information, Information and Interaction for Learning, Culture, Collaboration and Business, HIMI 2013, Lecture Notes in Computer Science*, Vol. 8018, Springer, Berlin, 2013, pp. 495-500.

[32] J. Kim, U. Yun, G. Pyun, H. Ryang, G. Lee, E. Yoon, K. H. Ryu, A Blog Ranking Algorithm Using Analysis of Both Blog Influence and Characteristics of Blog Posts, *Cluster Computing*, Vol. 18, No. 1, pp. 157-164, March, 2015.

[33] J.-H. Kim, T.-B. Yoon, K.-S. Kim, J.-H. Lee, Trackback-Rank: An Effective Ranking Algorithm for the Blog Search, *Proceedings of the 2008 Second International Symposium on Intelligent Information Technology Application - Volume 03*, Shanghai, China, 2008, pp. 503-507.

[34] R. Klein, A. Kyrilov, M. Tokman, Automated Assessment of Short Free-Text Responses in Computer Science Using Latent Semantic Analysis, *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education*, Darmstadt, Germany, 2011, pp. 158-162.

[35] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.

[36] F.-F. Kuo, M.-K. Shan, S.-Y. Lee, Background Music Recommendation for Video Based on Multimodal Latent Semantic Analysis, *Proceedings of the 2013 IEEE International Conference on Multimedia and Expo*, San Jose, CA, 2013, pp. 1-6.

[37] T. K. Landauer, D. S. McNamara, S. Dennis, W. Kintsch, *Handbook of Latent Semantic Analysis*, Psychology Press, 2013.

[38] M. H. C. Law, M. A. T. Figueiredo, A. K. Jain, Simultaneous Feature Selection and Clustering Using Mixture Models, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 9, pp. 1154-1166, September, 2004.

[39] H. Li, G. Ye, J. Wang, A Modified Algorithm for the Improvement of Composite Interval Mapping, *Genetics*, Vol. 175, No. 1, pp. 361-374, January, 2007.

[40] M. Li, W. K. Li, G. Li, On Mixture Memory Garch Models, *Journal of Time Series Analysis*, Vol. 34, No. 6, pp. 606-624, November, 2013.

[41] M. Lintean, C. Moldovan, V. Rus, D. McNamara, The Role of Local and Global Weighting in Assessing the Semantic Similarity of Texts Using Latent Semantic Analysis, *Proceedings of the 23th International Florida Artificial Intelligence Research Society Conference*, Daytona Beach, FL, 2010, pp. 235-240.

[42] C.-J. Luh, S.-A. Yang, D. T.-L. Huang, Estimating Search Engine Ranking Function with Latent Semantic Analysis and a Genetic Algorithm, *Proceedings of the 2012 3rd International Conference on E-Business and E-Government*, Shanghai, China, 2012, pp. 439-442.

[43] J. McInerney, A. Rogers, N. R. Jennings, Improving Location Prediction Services for New Users with Probabilistic Latent Semantic Analysis, *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, Pittsburgh, Pennsylvania, 2012, pp. 906-910.

[44] A. Mesaros, T. Heittola, A. Klapuri, Latent Semantic Analysis in Sound Event Detection, *Proceedings of the 19th European*

Signal Processing Conference, Barcelona, Spain, 2011, pp. 1307-1311.

[45] G. Mishne, M. de Rijke, A Study of Blog Search, in: M. Lalmas, A. MacFarlane, S. Rüger, A. Tombros, T. Tsikrika, A. Yavlinsky (Eds.), *Advances in Information Retrieval, ECIR 2006, Lecture Notes in Computer Science*, Vol. 3936, Springer, Berlin, 2006, pp. 289-301.

[46] B. A. Nardi, D. J. Schiano, M. Gumbrecht, L. Swartz, Why We Blog, *Communications of the ACM*, Vol. 47, No. 12, pp. 41-46, December, 2004.

[47] D. E. O'Leary, Blog Mining-Review and Extensions: From Each According to His Opinion, *Decision Support Systems*, Vol. 51, No. 4, pp. 821-830, November, 2011.

[48] M. G. Ozsoy, F. N. Alpaslan, I. Cicekli, Text Summarization Using Latent Semantic Analysis, *Journal of Information Science*, Vol. 37, No. 4, pp. 405-417, August, 2011.

[49] M. Pajovic, D. S. Millar, T. Koike-Akino, K. Kojima, V. Arlunno, K. Parsons, Multi-Pilot Aided Carrier Phase Estimation for Single Carrier Coherent Systems, *Proceedings of the 2015 Signal Processing in Photonic Communications*, Boston, Massachusetts, 2015, pp. SpT4D.4.

[50] Z. A. Pardos, N. T. Heffernan, Navigating the Parameter Space of Bayesian Knowledge Tracing Models: Visualizations of the Convergence of the Expectation Maximization Algorithm, *Proceedings of the third International Conference on Educational Data Mining*, Pittsburgh, PA, 2010, pp. 161-170.

[51] L. A. F. Park, K. Ramamohanarao, Efficient Storage and Retrieval of Probabilistic Lantent Semantic Information for Information Retrieval, *The VLDB Journal*, Vol. 18, No. 1, pp. 141-155, January, 2009.

[52] Pingdom, 2015 the Web Shown in Numbers!, from http://fast-traffic-seo.blogspot.tw.

[53] M. Porter, R. Boulton, Snowball: A Language for Stemming Algorithms, from http://snowball.tartarus.org/.

[54] Prayiush, Number of Blogs up from 35 Million in 2006 to 181 Million by the End of 2011, http://tinyurl.com/q9unvyy.

[55] B. Schwartz, Google Blog Search Now within Google News Search, http://searchengineland.com/google-blog-search-now-within-google-news-search-202202.

[56] J.-R. Shieh, C.-Y. Lin, S.-X. Wang, J.-L. Wu, Relational Term-Suggestion Graphs Incorporating Multipartite Concept and Expertise Networks, *ACM Transactions on Intelligent Systems and Technology*, Vol. 5, No. 1, pp. Article No. 19, December, 2013.

[57] A. Siddiqui, N. Mishra, J. S. Verma, A Survey on Automatic Image Annotation and Retrieval, *International Journal of Computer Applications*, Vol. 118, No. 20, pp. 27-32, May, 2015.

[58] J. Simões, F. Azeiteiro, J. Bernardino, Wise Blogs: A Special Blog Search Engine, *Proceedings of the Eighth International C* Conference on Computer Science & Software Engineering*, Yokohoma, Japan, 2015, pp. 144-145.

[59] Y. Takama, T. Kajinami, A. Matsumura, Blog Search with Keyword Map-Based Relevance Feedback, in: L. Wang, Y. Jin (Eds.), *Fuzzy Systems and Knowledge Discovery. FSKD 2005, Lecture Notes in Computer Science*, Vol. 3614, Springer, Berlin, 2005, pp. 1208-1215.

[60] T. Temel, A New Digital Cochlea Model Neuro-Spike Representation of Auditory Signals and Its Application to Classification of Bat-Like Biosonar Echoes, *Neural Network World*, Vol. 20, No. 2, pp. 223-239, May, 2010.

[61] M. Thelwall, L. Hasler, Blog Search Engines, *Online Information Review*, Vol. 31, No. 4, pp. 467-479, February, 2007.

[62] K.-P. Tung, W.-Z. Shi, R. De Silva, E. Edwards, D. Rueckert, Automatical Vessel Wall Detection in Intravascular Coronary Oct, *Proceedings of the 2011 Biomedical Imaging: From Nano to Macro*, Chicago, IL, 2011, pp. 610-613.

[63] W.-L. Zhang, B.-C. Li, W.-Q. Zhang, Compact Acoustic Modeling Based on Acoustic Manifold Using a Mixture of Factor Analyzers, *Proceedings of the 2013 Automatic Speech Recognition and Understanding*, Olomouc, Czech Republic, 2013, pp. 37-42.

[64] Wikipedia, Google Blog Search, https://en.wikipedia.org/wiki/Google_Blog_Search.

[65] WordNet, Wordnet Statistics, http://wordnet.princeton.edu/wordnet/man/wnstats.7WN.html.

[66] J. Xu, G. Ye, Y. Wang, G. Herman, B. Zhang, J. Yang, Incremental Em for Probabilistic Latent Semantic Analysis on Human Action Recognition, *Proceddings of the 6th IEEE International Conference on Advanced Video and Signal Based Surveillance*, Genova, Italy, 2009, pp. 55-60.

[67] G.-R. Xue, W. Dai, Q. Yang, Y. Yu, Topic-Bridged Plsa for Cross-Domain Text Classification, *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Singapore, 2008, pp. 627-634.

[68] J. Zhang, S. Gong, Action Categorization by Structural Probabilistic Latent Semantic Analysis, *Computer Vision and Image Understanding*, Vol. 114, No. 8, pp. 857-864, August, 2010.

[69] L. Zhu, A. Sun, B. Choi, Online Spam-Blog Detection through Blog Search, *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, Napa Valley, CA, 2008, pp. 1347-1348.

Biography

Lin-Chih Chen is an associate professor in the Department of Information Management at National Dong Hwa University, Taiwan. His research interests include Web Intelligent and Web Technology. He develops many Web Intelligent systems include Cayley search engine, On-The-Fly Document Clustering, LI keyword suggestion system, WSC clustering system, Cayley Scholar.