Finding Relevant Concepts for Unknown Terms Using a Web-based Approach

Chen-Ming Hung[1] and Lee-Feng Chien[1, 2]

1. *Institute of Information Science, Academia Sinica*
2. *Dept. of Information Management, National Taiwan University*
Taipei, Taiwan

rglly@iis.sinica.edu.tw and lfchien@iis.sinica.edu.tw

Abstract. Previous research on automatic thesaurus construction most focused on extracting relevant terms for each term of concern from a small-scale and domain-specific corpus. This study emphasizes on utilizing the Web as the rich and dynamic corpus source for term association estimation. In addition to extracting relevant terms, we are interested in finding concept-level information for each term of concern. For a single term, our idea is that to send it into Web search engines to retrieve its relevant documents and we propose a Greedy-EM-based document clustering algorithm to cluster them and determine an appropriate number of relevant concepts for the term. Then the keywords with the highest *weighted log likelihood ratio* in each cluster are treated as the label(s) of the associated concept cluster for the term of concern. With some initial experiments, the proposed approach has been shown its potential in finding relevant concepts for unknown terms.

1. INTRODUCTION

It has been well recognized that a thesaurus is crucial for representing vocabulary knowledge and helping users to reformulate queries in information retrieval systems. One of the important functions of a thesaurus is to provide the information of term associations for information retrieval systems. Previous research on automatic thesaurus construction most focused on extracting relevant terms for each term of concern from a small-scale and domain-specific corpus. In this study, there are several differences extended from the previous research. First, this study emphasizes on utilizing the Web as the rich and dynamic corpus source for term association estimation. Second, the thesaurus to be constructed has no domain limitation and is pursued to be able to benefit Web information retrieval, e.g. to help users disambiguate their search interests, when users gave poor or short queries. Third, in addition to extracting relevant terms, in this study we are interested in finding concept-level information for each term of concern. For example, for a term "National Taiwan University" given by a user, it might contain some different but relevant concepts from users' point of view, such as "main page of National Taiwan University", "entrance examination of NTU", "the Hospital of NTU", etc. The purpose of this paper is, therefore, to develop an efficient approach to deal with the above problem.

In information retrieval researching area, extracting concepts contained in one text always plays a key role. However, in traditional way, if the text is too short, it is almost impossible to get enough information to extract the contained concepts. In this paper, utilizing the abundant corpora on the World Wide Web, we attempt to find the concepts contained in arbitrary length of topic-specific texts, even only a single term. For a single term, our idea is that to send it into Web search engines to retrieve its relevant documents, and a Greedy-EM-based document clustering algorithm is developed to cluster these documents into an appropriate number of concept clusters, with the similarity of the documents. Then the terms with the *highest weighted log likelihood ratio* in each clustered document group are treated as the label(s) of the associated concept cluster for the term of concern. To cluster the extracted documents into an unknown number of concept mixtures is important, because it is hard to know an exact number of concepts should be contained in a single term.

Compared with general text documents, a single term is much shorter and typically do not contain enough information to extract adequate and reliable features. To assist the relevance judgment between short terms, additional knowledge sources would be exploited. Our basic idea is to exploit the Web. Adequate contexts of a single term, e.g., the neighboring sentences of the term, can be extracted from large amounts of Web pages. We found that it is convenient to implement our idea using the existent search engines. A single term could be treated as a query with a certain search request. And its contexts are then obtained directly from the highly ranked search-result snippets, e.g., the titles and descriptions of search-result entries, and the texts surrounding matched terms.

The proposed approach relies on an efficient document clustering technique. Usually, in document clustering techniques [1, 3, 4, 8], each text in a training set is transformed to a certain vector space, then begin agglomerated with another one text step by step depending on their cosine similarity [9]. Thus, a proper

transformation from text to vector space, like TFIDF [9], takes the heavy duty of classification accuracy or concept extraction result. However, a good transformation, i.e. feature extraction, needs a well-labeled training data to support; this is not such an easy task in real world. The idea of this paper is to modify the vector space transformation as probabilistic framework.

With the extracted training data from the web, the Greedy EM algorithm [5, 7] is applied in this paper to automatically determine an appropriate number of concepts contained in the given single term through clustering the training documents. This is important while doing relevant concept extraction; otherwise, the number of concepts has to be assumed previously, it is difficult and impractical in real world. After clustering the training documents extracted from the Web into a certain number of mixtures, for each mixture, the representation of this mixture is straightforwardly defined as the term with the highest *weighted log likelihood ratio* in this mixture. With some initial experiments, the proposed approach has been shown its potential in finding relevant concepts for terms of concern.

The remainder of the paper is organized as follows. Section 2 briefly describes the background assumption, i.e. Naïve Bayes, and the modeling based on Naive Bayes. Section 3 describes the overall proposed approach in this paper, including the main idea of the greedy EM algorithm and its application to decide the number of concept domains contained in the training data from the web; in addition, generates keywords via comparing the *weighted log likelihood ratio*. Section 4 shows the experiments and their result. The summary and our future work are described in Section 5.

2. NAÏVE BAYES ASSUMPTION AND DOCUMENT CLASSIFICATION

Before introducing our proposed approach, here introduce a well known way of text representation, i.e., Naive Bayes assumption. Naive Bayes assumption is a particular probabilistic generative model for text. First, introduce some notation about text representation. A document, d_i, is considered to be an ordered list of words, $\{w_{d_{i,1}}, w_{d_{i,2}}, \mathbb{K}\ , w_{d_{i,|d_i|}}\}$, where $w_{d_{i,j}}$ means the jth words and $|d_i|$ means the number of words in d_i. Second, every document is assumed generated by a mixture of components $\{C_k\}$ (relevant concept clusters), for k=1 to K. Thus, we can characterize the likelihood of document d_i with a sum of total probability over all mixture components:

$$p(d_i \mid \theta) = \sum_{k=1}^{K} p(C_k \mid \theta)\, p(d_i \mid C_k, \theta) \qquad (1)$$

Furthermore, for each topic class C_k of concern, we can express the probability of a document as:

$$\begin{aligned} p(d_i \mid C_k, \theta) &= p(< w_{d_{i,1}}, w_{d_{i,2}}, \mathbb{K}\ , w_{d_{i,|d_i|}} > \mid C_k, \theta) \\ &= \prod_{j=1}^{|d_i|} p(w_{d_{i,j}} \mid C_k, \theta, w_{d_{i,z}}, z < j) \end{aligned} \qquad (2)$$

$$p(w_{d_{i,j}} \mid C_k, \theta, w_{d_{i,z}}, z < j) = p(w_{d_{i,j}} \mid C_k, \theta) \qquad (3)$$

Based on standard Naive Bayes assumption, the words of a document are generated independently of context, that is, independently of the other words in the same document given the class model. We further assume that the probability of a word is independent of its position within the document. Combine (1) and (2),

$$p(d_i \mid C_k, \theta) = \prod_{j=1}^{|d_i|} p(w_{d_{i,j}} \mid C_k, \theta) \qquad (4)$$

Thus, the parameters of an individual class are the collection of word probabilities, $\theta_{w_t|C_k} = p(w_t \mid C_k, \theta)$. The other parameters are the weight of mixture class, $p(C_k \mid \theta)$, that is, the prior probabilities of class, C_k. The set of parameters is $\theta = \{\theta_{w_t|C_k}, \theta_{C_k}\}$. As will be described in next section, the proposed document clustering is designed fully based on the parameters.

3. RELEVANT CONCEPTS EXTRACTION

In this section, we describe the overall framework of the proposed approach. Suppose given a single term, T, and its relevant concepts are our interest. The first step of the approach is to send *T* into search engines to retrieve the relevant documents as the corpus. Note that the retrieved documents are the so-called snippets defined in [2]. The detailed process of the approach is described below.

3.1 The proposed Approach

Suppose given a single term, *T*; then the process of relevant-concept extractions is designed as:

Step 1. Send *T* into search engines to retrieve *N* snippets as the Web-based corpus, D_T.

Step 2. Apply the Greedy EM algorithm to cluster D_T into *K* mixtures (clusters), $\{C_k\}_{k=1}^{K}$, where *K* is dynamically determined.

Step 3. For each C_k, k=1 to *K*, choose the term (s) with the highest *weighted log likelihood ratio* as the label (s) of C_k.

3.2 The Greedy EM Algorithm

Because we have no idea about the exact number of concepts strongly associated with each given term, thus for each term it's straightforward to apply the Greedy EM algorithm to clustering the relevant documents into an auto-determined number of clusters. The algorithm is a top-down clustering algorithm which is based on the assumptions of the theoretical evidence developed in [5, 7]. Its basic idea is to suppose that all the relevant documents belong to one component (concept cluster) at the initial stage, then successively adding one more component (concept cluster) and redistributing the relevant documents step by step until the maximal likelihood is approached.

Figure 1 shows the proposed approach and it is summarized in the following.

a) Set *K*=1 and initialize $\theta_{C_k} = 1$ and $\theta_{w_t|C_k}$ straightforwardly by w_t's frequency, for all w_t shown in D_T.

b) Perform EM steps until convergence, then $\theta^i = \{\theta_{w_t|C_k}, \theta_{C_k}\}_{k=1}^{K}$

c) Calculate the likelihood, $L(\theta^i)$.

d) Allocate one more mixture given initial modeling, i.e. $\theta_{K+1} = \{\theta_{w_t|C_{K+1}}, \theta_{C_{K+1}}\}$, described in section 3.2.2.

e) Keep θ^i fixed, and use partial EM techniques, described in section 3.2.3, to update θ_{K+1}.

f) Set $\theta^{i+1} = \{\theta_{w_t|C_k}, \theta_{C_k}\}_{k=1}^{K+1}$. Calculate the likelihood, $L(\theta^{i+1})$.

g) Stop if $L(\theta^{i+1}) < L(\theta^i)$; otherwise, return to c) and set *K*=*K*+1.

3.2.1 Likelihood Function

As described previously, all relevant documents belong to one mixture initially; then check the likelihood to see if it is proper to add a new mixture. Thus, given *K* mixture components, the likelihood of *K*+1 is defined as:

$$L_{K+1}(D_T) = (1-\alpha)L_K(D_T) + \alpha\phi(D_T, \theta_{K+1}) \qquad (5)$$

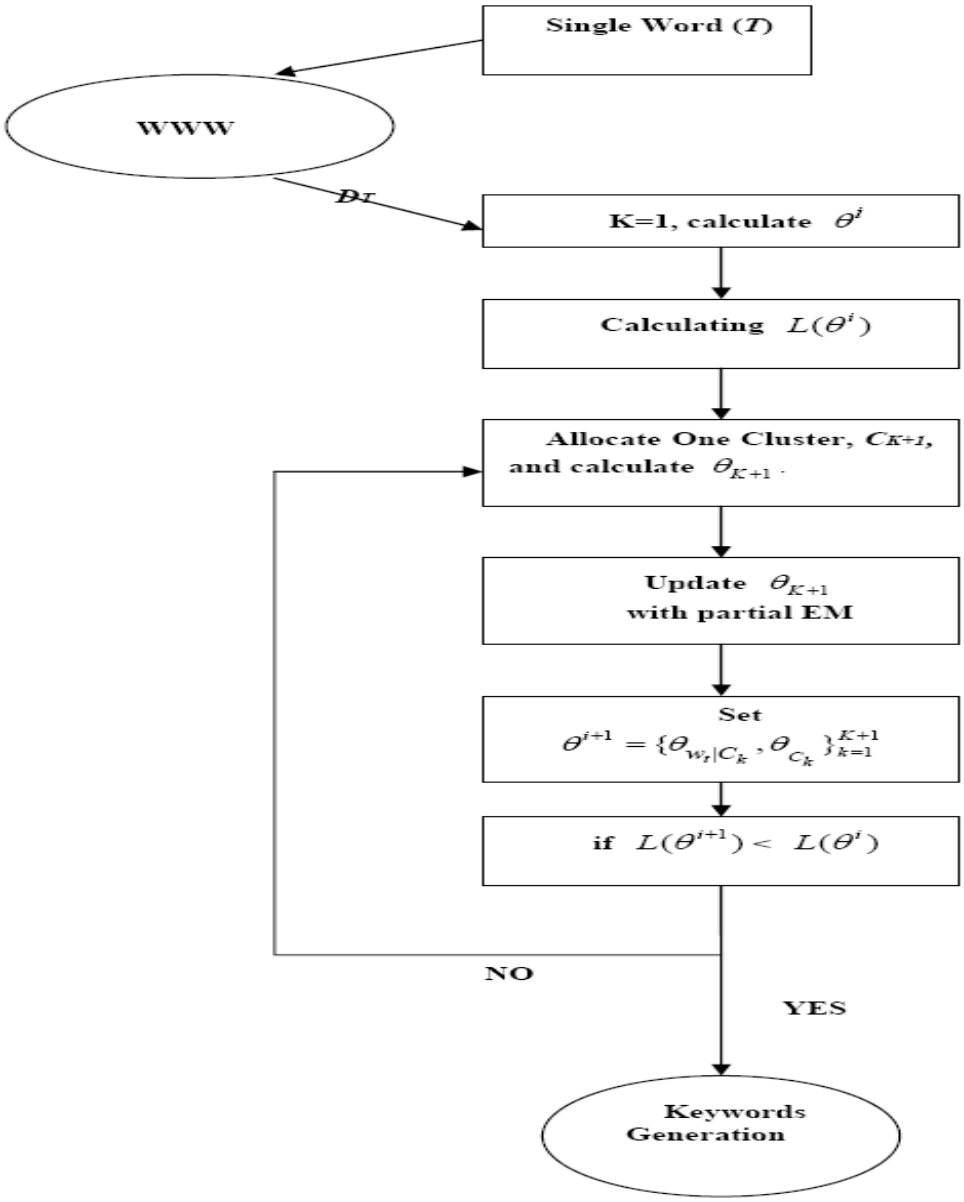

Figure 1. Overall Framework Proposed

with α in (0,1), where $\theta_{K+1} = \{\theta_{w_t|C_{K+1}}, \alpha\}$ is the modeling of newly added mixture C_{K+1} and $\phi(D_T, \theta_{K+1})$ is the likelihood in C_{K+1}. If $L_{K+1}(D_T) < L_K(D_T)$, then stop the allocation of new mixture; otherwise, reallocate a new one.

3.2.2 Initialize Allocated Mixture

In [7], a vector space model, initializing the newly added mixture is to calculate the first derivation with respect to α and to assume that the covariance matrix is a constant matrix. However, in our proposed probability framework, it is much more complicated because of a large amount of word probabilities, $\{\theta_{w_t|C_k}\}$ $\forall w_t$. Thus, we take the approximation of α in [6] as $\alpha = 0.5$ for K=1 and $\alpha = 2/(K+1)$ for $K \geq 2$. The initialization of $\{\theta_{w_t|C_{K+1}}\}$ $\forall w_t$ is randomized to satisfy $\sum_{\{w_t\}} \theta_{w_t|C_k} = 1$.

3.2.3 Update with Partial EM Algorithm

In order to simplify the updating problem, we take advantage of partial EM algorithm for locally search the maxima of $L_{K+1}(D_T)$. A notable property is that the original modeling for k=1 to K are fixed, only θ_{K+1} is updated.

$$\theta_{w_t|C_{K+1}} = \{p(w_t \mid C_{K+1})\}_{t=1}^{|V|} = \left\{ \frac{1+\sum_{n=1}^{|D_T|} N(w_t, d_n)\, p(C_{K+1} \mid d_n)}{|V| + \sum_{s=1}^{|V|}\sum_{n=1}^{|D_T|} N(w_s, d_n)\, p(C_{K+1} \mid d_n)} \right\}_{t=1}^{|V|} \quad (6)$$

$$\theta_{C_K+1} = p(C_{K+1}) = \frac{1+\sum_{n=1}^{|D_T|} p(C_{K+1} \mid d_n)}{(K+1)+|D|} \quad (7)$$

where $|V|$ and $|D_T|$ means the number of vocabularies and the number of documents shown in the D_T respectively.

Since only the parameters of the new components are updated, partial EM steps constitute a simple and fast method for locally searching for the maxima of L_{K+1}, without needing to resort to other computationally demanding nonlinear optimization methods.

3.3 Keyword Generation

In the process, the documents in the training data D_T will be clustered with their similarity into a set of clusters and keywords that can represent the concept of each cluster will be extracted. After clustering the relevant documents into several clusters, the distribution of each cluster in a probabilistic form can be calculated with the data in the cluster by applying the Greedy EM algorithm already described previously.

Next, we have to discover the hidden semantics inside each document cluster. However, retrieving the hidden semantics from a set of documents is a big issue. For convenience, we simply represent the meaning of a cluster with the word that has the highest *weighted log likelihood ratio*[1] among the contained words in this cluster. With this assumption, the "representative" word could be chosen directly by comparing

$$WLR(w_t \mid C_k) = p(w_t \mid C_k) \log\left(\frac{p(w_t \mid C_k)}{p(w_t \mid \overline{C}_k)} \right) \quad (8)$$

for k=1 to K, where $p(w_t \mid C_k)$ means the probability of word w_t in component C_k and $p(w_t \mid \overline{C}_k)$ means sum of the probabilities of word w_t in those clusters except C_k.

4. EXPERIMENTS

In real world, for an unknown term, its associated concepts are what we are interested in; thus, in this section, we will show the experiment results obtained in evaluating a set of test terms. Before the larger amount of experiment, let's preview the experiment of "ATM" to determine the number of retrieved relevant documents. Google (http://www.google.com) is the main search engine which we utilized in the following experiment.

4.1 Appropriate Number of Retrieved Relevant Documents

[1] The sum of this quantity over all words is the Kullback-Leibler divergence between the distribution of words in C_k and the distribution of words in $\overline{C}_k$, (Cover and Thomas, 1991).

We assume that too many retrieved documents will cause noises, but too few won't contain enough information about this unknown term. Thus, the appropriate number of retrieved relevant documents has to be decided. “ATM” in dictionary has six hidden semantics, which are “Automated Teller Machine”, “Asynchronous Transfer Mode”, “Act of Trade Marks”, “Air Traffic Management”, “Atmosphere” and “Association of Teachers of Mathematics” respectively. Table 1 shows the bi-gram extracted concepts via number of retrieved texts.

Table 1: *Extracted concept clusters in “ATM” with respect to different numbers of retrieved relevant terms*

# of training texts	Extracted concept clusters
100	*ATM {card, cell, internetworking, standards}, asynchronous transfer*
200	*ATM {access, information, networking, standards, locations}, credit union, debit cards , safety tips, teller machine, telescope makers*
300	*ATM {applications, cards, fees, networking, surcharges, locations}, adaptation layers, credit union, debit cards, token rings, teller machine,*
400	*ATM {applications, crashes, services, transactions, networking, security}, branch locator, financial institution, personal banking, public transport, telangiectasia mutated*
500	*ataxia telangiectasia, ATM {applications, asynchronous, crashes, products, protocol, resource}*
600	*ataxia telangiectasia, ATM {applications, crashes, protocol, technology}, atmospheric science, communication technology, electronics engineering, network interface, public transport, wan switches, rights reserved*
700	*air traffic, ataxia telangiectasia, ATM {crashes, debit, encryptor, protocol, surcharge, traffic}, atmospheric science, checking account, communication technology, electronics engineering, network interface, public transport*
800	*ataxia telangiectasia, ATM {adapters, crime, debit, protocol, surcharge, usage, cards}, atmospheric sciences, business checking, communication technology*
900	*24 hours, ataxia telangiectasia, ATM {adapters, connections, crashes, crime, debit, industry, protocol, resources}*
1000	*ATM networks, Arizona federal, 24 hour*

Table 1 shows a challenge that choosing the term with the highest *weighted log likelihood ratio* as the label of one concept cluster can not effectively describe its complete semantics appropriately; in addition, for example, “Automated Teller Machine” is composed of many aspects, like security, location, cards, and etc. Thus, concept domain of “Automated Teller Machine” could be figured out while “ATM applications”, “ATM locations”, “ATM surcharges”, and some other aspects associated with “Automated Teller Machine” being extracted. Similarly, “Air Traffic Management” could be figured out while “public transport”, “air traffic” being extracted.

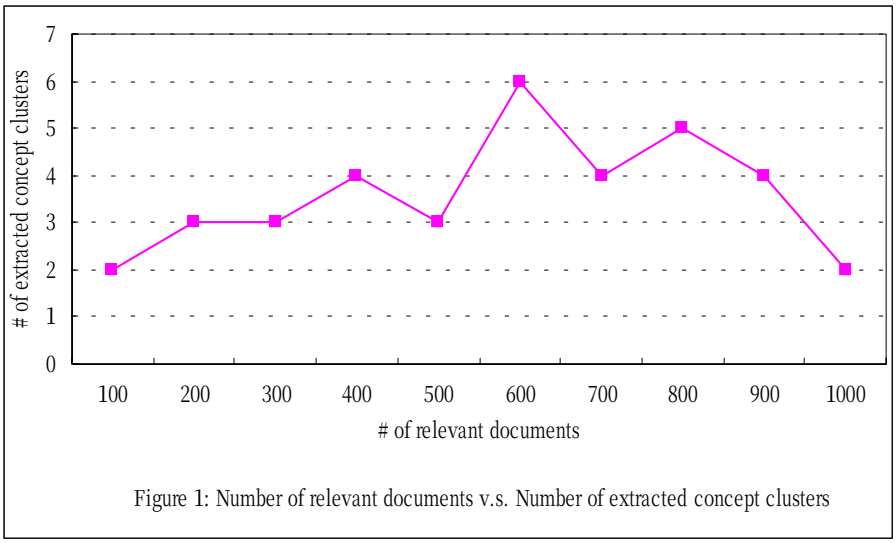

Figure 1: Number of relevant documents v.s. Number of extracted concept clusters

Except the six hidden semantic clusters in ATM, some other concept clusters were also extracted, e.g. “Amateur Telescope Maker” because of “telescope makers” extracted and “Ataxia Telangiectasia Mutated” because of “ataxia teleangiectasia” extracted. One more interesting thing is that the more retrieved relevant documents not necessarily direct to the more extracted concept clusters (Figure 1). This phenomenon is caused from the extra noises extracted from the more relevant documents. The extra noises will not only worsen the performance of the greedy EM algorithm but also generate improper relevant terms from the clustered groups, which will not be considered as “good” categories manually. For each test term, considering the time cost and the marginal gain of extracted concepts, 600 relevant documents were retrieved from Web. Of course 600 relevant documents are not always appropriate for all cases, but for convenience, it was adopted.

4.2 Data Description

The experiments took the "Computer Science" hierarchy in Yahoo! as the evaluation. There were totally 36 concepts in second level in the "Computer Science" hierarchy (as in Table 2), 177 objects in the third level and 278 objects in fourth level, all rooted at the concept "Computer Science". We divided the objects in third-level and fourth-level into three groups: full articles, which were the Web pages linked from Yahoo!'s website list under the Computer Science hierarchy, short documents, which were the site description offered by Yahoo!, and text segments, which were the directory names. We randomly chose 30 text segments from the third-level plus the fourth-level objects. The 30 proper nouns are shown in Table 3.

Table 2: "Computer Science" hierarchy in Yahoo!

algorithms	library and information science
architecture	linguistics
artificial intelligence	logic programming
compression	mobile computing
computational learning theory	modeling
computational sciences	networks
computer vision	neural networks
databases	objective oriented programming
distributed computing	operating systems
DNA-based computing	quantum computing
electronic computer aided design	real time computing
end user programming	robotics
finite model theory	security and encryption
formal methods	software engineering
graphics	supercomputing and parallel computing
handwriting recognition	symbolic computation
human computer interaction	user interface
knowledge sciences	virtual reality

4.3 Relevant-Concepts Extraction

In Section 3, the Greedy EM algorithm is treated as the unsupervised learning method which clusters retrieved relevant documents to extract hidden concepts for each test term.

Table 3: *30 terms from 3rd level and 4th level in Yahoo!’s CS hierarchy*

ActiveX	IRIX	ROADS
CMX	ISDN	RSA
CORBA	Jini	Ray Tracing
Darwin	JXTA	SETL
Ebonics	Mach	SIP
Eponyms	Mesa	Trigonometry
Figlet	PGP – Pretty Good Privacy	VHDL
GNU	PPP	VMS
Hobo Signs	Puns	WAIS
Hurd	QNX	Xinu

Table 4 shows the extracted bi-gram concept clusters for the 30 randomly chosen CS terms; this means that only bi-gram terms in the retrieved documents were extracted. The number of hidden concept clusters in each term was determined automatically by the Greedy EM algorithm.

Table 4: B*i-gram concept clusters for the test terms in Yahoo!'s CS hierarchy*

Test Terms	**Extracted Concept Clusters**
ActiveX	*ActiveX {control, vs, server}*
CMX	*CMX-RTX RTOS, multi-tasking operating, CMX {3000, 5000}, San Jose, Jose BLVD*
CORBA	*application development, C++ software, CORBA {2.2, orbs, servers}, distributed {applications, programming}, IDL compiler, language {IDL, mapping}, object-oriented programming, request broker*
Darwin	*Charles Darwin, Darwin 6.0.2*
Ebonics	*black English, African Americans, Ebonics X-mas*
Eponyms	*medical phenomena, on-line medical, aortic regurgitation, encyclopediaof medical, Firkin Judith, esophageal surgery, historical allusions*
Figlet	*art characters, Figlet {Frank, frontend, package, RPM, tool}, assorted fonts*
GNU	*GNU {aspell, coding, compilers, documentation, desktop}, license GPL, public licenseterms, reference card*
Hobo Signs	*ideogram carved, 45 signs*
Hurd	*Alexander Hurd, Debian developers, Debian Gnu, GNU operating, Hannah Hurd, Hon Lord*
IRIX	*Sgi IRIX, IRIX reinstall, 2.6.5.7 Sgi, 3D graphics*
ISDN	*digital {access, networks, telephone}, Arca technologies, communication standards, copper wire, data {applications, communications, services}, external ISDN, integrated services*
Jini	*Jini technology, Jini Ji*
JXTA	*project JXTA, peer peer*
Mach	*Mustang Mach, disc golf*
Mesa	*Mesa Verde, Mesa Quad*
PGP	*encrypt messages, foaf files, keysigning party, PGP {backend, basics, comments, corp}, ASCII armour*
PPP	*point-to-point protocol, ppp flea.,*
Puns	*French word, Japanese spelling, social sciences, bilingual puns*
QNX	*Microkernel OS, QNX {applications, machine, voyager}, alternative vendor*
ROADS	*{Access, British, Hampton} ROADA, adverse weather*
RSA	*RSA security, RSA lighting*
Ray Tracing	*Computer graphics, Carlo Ray, recursive Ray*
SETL	*set language, ab le*
SIP	*control protocol, {bring, panel, partysip,} SIP, SIP {architecture, application, client, standards}, Jonathan Rosenberg*
Trigonometry	*Trigonometric functions, advanced algebra, Benjamin Bannekers, Banneker's trigonometry*
VHDL	*Asic design, circuit VHSIC, complete VHDL, hardware design, digital logic, verilog simulation, synthesis tool*
VMS	*computational chemistry, shopping cart, administrator authentication, UNIX translation, CCL VMS, equipment corporation*
WAIS	*area informationserver, laws enacted, public laws, WAIS {client, gateway, searching, libraries}, presidential documents*
Xinu	*AMD élan, unix clone, software OS, Xinu {kernel, system}, II internetworking, master distributor, demand paging*

From Table 4, it is encouraging that the proposed approach extracted the main idea for most test CS terms. Taking "Trigonometry" for example, if we have no idea about "Trigonometry", then from "function" and "algebra" in Table 4, there is not difficult to guess that it may be a kind of mathematical functions and

developed by Benjamin Bannekers. Again, our proposed approach caught that “Darwin” is not only a British Naturalist but also the name of graphical software.

Even though the experiment result shows encouraging performance, the result was still bothered by many duplicated and noisy aspects. For example, “CORBA” means “Common Object Request Broker Architecture”; however, “C++ software” and “application development” actually only provide vague or not necessary information about “CORBA”. This was caused by the “too much effort” of the Greedy EM algorithm, which clusters the retrieved mixtures into too many groups.

5. CONCLUSIONS AND FUTURE WORK

We have presented a potential approach to finding relevant concepts for terms via utilizing World Wide Web. This approach obtained an encouraging experimental result in testing Yahoo!’s computer science hierarchy. However, the work needs more in-depth study. As what we mentioned previously, choosing the word with the highest *weighted log likelihood ratio* as the concept of a clustered group after the Greedy EM algorithm does not provide enough representative. In addition, one concept usually contains many domains, e.g. “ATM” contains security, teller machine, transaction cost, and etc. Thus, distinguishing the extracted keywords into a certain concept still needs human intervention. On the other hand, in order to solve the problem of “too much effort” of the Greedy EM algorithm, we need to modify it with another convergence criterion.

References

[1] A. Jain, M. Murty, and P. Flynn. Data Clustering: A Review. In ACM Computing Surveys, 31(3), September 1999.

[2] C. C. Huang, S. L. Chuang and L. F. Chien. LiveClassifier: Creating Hierarchical Text Classifiers through Web Corpora, *WWW* (2004).

[3] E. Rasmussen. Clustering Algorithms. In *Information Retrieval Data Structures and Algorithms*, William Frakes and Ricardo Baeza-Yates, editors, Prentice Hall, 1992.

[4] E. Voorhees. The Cluster Hypothesis Revisited. In *Proceedings of SIGIR* 1985, 95-104.

[5] J. J. Verbeek, N. Vlassis and B. J. A. Krose. Efficient Greedy Learning of Gaussian Mixture Models. *Neural Computation*, 15 (2), pp.469-485, 2003.

[6] J. Q. Li and A. R. Barron. Mixture Density Estimation. In *Advances in Neurarl Information processing Systems* 12, The MIT Press, 2000.

[7] N. Vlassis and A. Likas A Greedy Algorithm for Gaussian Mixture Learning. In *Neural Processing Letters* (15), pp. 77-87, 2002.

[8] P. Willett. Recent Trends in Hierarchic Document Clustering: A Critical Review. In *Information Processing and Management*, 24(5), 577-597, 1988.

[9] T. Joachims. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In *Machine Learning: Proceedings of the Fourteenth International Conference (ICML ’97)*, pp. 143-151.