Hierarchical Web Document Classification Based on Hierarchically Trained Domain Specific Words

Jing-Shin Chang
Department of Computer Science & Information Engineering
National Chi-Nan University
1, University Road, Puli, Nantou, Taiwan, ROC.

jshin@csie.ncnu.edu.tw

Abstract

Search engines return thousands of pages per query. Many of them are relevant to the "query words" but not interesting to the "users" due to different domain-specific meanings of the query terms. Re-classification of the returned documents based on domain specific meanings of the query terms would therefore be most effective. A *cross domain entropy (CDE)* measure is proposed to extract characteristic domain specific words (DSW's) for each node of existing hierarchical web document trees. Domain specific class models are built based on the respective DSW's. Such class models are then used for directly classifying new documents into the hierarchy, instead of using hierarchical clustering techniques. High accuracy can be achieved with very few domain specific words. With only the top 5~10% DSW's and a *maximum entropy* based classifier, 99% accuracy is observed when classifying documents of a news web site into 63 domains. The precision and recall of the extracted domain specific words are also higher than those extracted with conventional TF-IDF term weighting method.

Keywords: Domain Specific Words, Hierarchical Classification, Maximum Entropy Classifier, Cross-Domain Entropy.

1 Re-classification Issues

Search engines today return thousands of pages per query. Many of them are relevant to the "query words" but not always interesting to the "users". The major problem is that search engines do not distinguish query terms with multiple word senses. For instance, given the query term "Jaguar", most search engines are unable to identify whether it refers to an animal, a car, or an air gunship (strike fighter plane). Given a user name, most search engines cannot distinguish the person as a teacher, a technician or a government officer either. Re-classification of the web documents, so that users can quickly identify the interested documents, is therefore highly desirable.

For easy access, hierarchical classification of documents into a well-justified document hierarchy would be the most appropriate [2, 3, 4, 5, 6]. By "well-justified", we mean a hierarchy that was created or customized by human web masters, instead of an artificial hierarchy created with some automatic clustering approach. An effective classifier and a set of discriminative features for word sense disambiguation (WSD) are the key components for such purposes.

Many methods have been proposed to extract useful "features" for building classification models [13]. For hierarchical classification into an existing document hierarchy, the most effective approach would be to extract discriminative domain specific words (DSW's) for each node of the hierarchy, and build a classification model for each node directly based on such words.

A method for learning domain specific words from a web hierarchy, building associated domain-specific class model, and then classifying the documents directly into such a hierarchical document tree is therefore a key issue for re-classification.

2 Domain Specific Words as Discriminative Feature Words for Document Classification

For effective word sense disambiguation (WSD) in the reclassification process, characteristic domain specific words (DSW's) associated with each node of the document hierarchy will play an important role. The existence of some DSW's in a document will be strong evidence for the document to be a specific class and for the embedded words to be of specific senses. For instance, the existence of the domain specific word "basketball" in a document will strongly suggest the "sport" class of the particular document, even though it might have been accessed by a query term like "Pistons" (which normally has a machinery sense). With the DSW "basketball", the special usage of Piston as a basketball team, rather than its "machinery" sense, are likely to be recognized. Actually, both of them act as domain specific words of the sport domain, and enhance each other for supporting the "sport" class when both appear in the same document. DSW identification is therefore an important issue for document re-classification.

However, manually acquiring the associated domain specific words for each node in the hierarchy is most likely unaffordable in terms of time and cost. Therefore, learning domain specific words automatically from existing web hierarchy is the key step for web document re-classification. In addition, new words (or new usages of words) are dynamically produced day by day. For instance, the Chinese word "活塞" (piston) is more frequently used as the "sport" or "basketball" sense (referring to the Detroit Pistons) in Chinese web pages rather than the "mechanics" or "automobile" sense. It is therefore desirable to find an automatic and inexpensive way to acquire the whole set of domain specific words of the hierarchy directly from web corpora.

Actually, the directory hierarchy of a Web can be regarded as a kind of classification tree for web documents. Each node of the hierarchy is associated with a special class label, identified by the directory name, and a large number of documents with some kind of domain specific words. For instance, the documents under the "sport" hierarchy are likely to use a large number of domain-specific words for the "sport" class. Hence, the domain specific words for each node can be *extracted by removing "non-specific terms"* from the associated document sets provided a measure of "domain specificity" is well established. A cross-domain entropy (CDE) measure will be proposed for this purpose.

With these DSW's, associated with each node, a class model for each such node can be established for direct classification of web documents into the document hierarchy.

3 Classification by Clustering vs. Direct Classification into Web Hierarchy

As indicated, for easy access, hierarchical classification of the documents into a well-justified document hierarchy is important. Automatic construction of the document hierarchy by document clustering, however, might create a hierarchy that is not acceptable by users.

The conventional clustering approach to classify web documents into a hierarchical structure is to collect "important terms" in all web documents as their representatives and measure the distance between document pairs using some well-known distance metrics between documents. For instance, the vector space model [10] measures the cosine of the angle between two document vectors, consisting of weights for important terms, as a similarity measure. Documents with high similarity or short distances are then clustered bottom-up to build a hierarchy. The clustering hierarchy is then manually inspected for adjustment into a customized hierarchy if necessary.

There are several disadvantages with this approach. First of all, since the documents are clustered based on distance or similarity measures that do not have a direct link with any ontological criteria, the hierarchical relationship among the clustered web documents is not guaranteed to fit any naturally created hierarchy by most web masters. In particular, most clustering algorithms merge documents in a binary way, which is far from the way a human user would do. The mismatch of such hierarchical structures implies that the clustered one might not match human perception quite well. The clustering results may therefore not be acceptable by the users. Furthermore, due to such mismatch, the clustered hierarchy may not be adjusted comfortably by the web masters. As far as the computation cost is concerned, computation of document distances based on *pairwise* distance metrics will be time consuming. (Admittedly, clustering might be preferred in some circumstances [4].)

Fortunately, clustering is not the only option since a large number of web documents, which are natively arranged in a *hierarchical* manner, are created every day. One can readily learn to classify documents directly into a customized hierarchy by learning the domain specific words of each node from the training documents associated with each node, and creating a class model for each node.

4 Building Domain Specific Classifier Models with Domain Specific Words Detected from Web Hierarchy

Since the web documents already form an extremely huge document classification tree, we propose here a simple and automatic approach to acquire the domain specific words in the hierarchical web documents. The domain specific words for each node can then be used to build their respective classifier models for each node of the document tree.

This simple approach is inspired by the fact that most text materials (webpages) in websites are already classified in a hierarchical manner; the hierarchical directory structures implicitly suggest that the domain specific terms in the text materials of a particular subdirectory are closely related to a common subject, which is identified by the name of the subdirectory. If we can detect domain specific words within each document, by removing

words that are non-specific to the subdirectory, then the resultant DSW's would be good representative for building classifier models for the subdirectory where they reside.

For instance, a subdirectory entitled 'entertainment' is likely to have a large number of web pages containing domain specific terms like 'singer', 'pop songs', 'rock & roll', 'album', and so on. Since these words are good representatives of the 'entertainment' domain, they can be used to build a class model for the 'entertainment' domain. A new document accessed by the query term 'album', as well as other domain specific terms, can then be classified into the 'entertainment' class easily, instead of into the 'photo album' class or something else.

Since a large number of documents had been classified into various web hierarchies, and update of the hierarchies is a daily routine of the various webmasters, the training corpora is not sparse. As such, we will pay almost no cost in training the classifier models for various domains and customized document trees.

The idea might extend equally well to other hierarchically organized Internet resources, such as news groups and BBS articles. Extending the idea to hierarchically organized book chapters might also be possible.

The advantages of building classification models directly from the sets of DSW's associated with each node, by removing non-specific terms from documents, are many folds. First, the original hierarchical structure reflects human perception on which directory a document should be classified into. Therefore, one rarely needs to adjust the hierarchy; in the worse case, one may be more comfortable to adjust the hierarchy if necessary. Second, the computation cost is significantly reduced, since pairwise computation of document distance is now replaced by the computation of "domain specificity" of documents against domains. The reduction is significant, from $O(\sum|d|x\sum|d|)$ to $O(\sum|d|x|D|)$, where $|d|$ and $|D|$ represent the number of documents in individual domains and number of domains, respectively.

5 Domain Specific Word Acquisition: A Cross-Domain Entropy Approach

Since the terms in the documents include general terms as well as domain-specific terms, the only problem then is an effective model to exclude those domain-independent terms from the documents. The degree of domain independency can be measured with the cross-domain entropy (CDE) as will be defined in the following DSW (Domain-Specific Word) Extraction Algorithm. Intuitively, a term that distributes evenly in all domains is likely to be independent of any domain and therefore is unlikely to be a DSW. The CDE provides a way to better estimate domain independency than traditional inverse document frequency (IDF). The method is first revealed in [7] and is summarized as follows.

First of all, a large collection of web documents is acquired using a web spider while preserving the directory hierarchy. Since our target is Chinese documents, a word segmentation algorithm [12, 9] is applied in order to identify terms in the documents.

For each subdirectory d_j, we find the number of occurrences n_{ij} of each term w_i in all the documents, and derive the normalized term frequency $f_{ij} = n_{ij} / N_j$ by normalizing n_{ij} with the total document size, $N_j \equiv \sum_i n_{ij}$, in that directory. The directory is then associated with a set of $< w_i, d_j, f_{ij} >$ tuples, where w_i is the i-th words of the complete word list for all documents, d_j is the j-th directory name (refer to as the domain hereafter), and $f_{ij} = n_{ij} / N_j$ is the normalized relative frequency of occurrence of w_i in domain d_j.

Domain-independency of the terms are then estimated with the following **Cross-Domain Entropy** (CDE) measure [7]:

$$H_i^* \equiv H^*(w_i) \equiv -\sum_j P_{ij} \log P_{ij}$$

$$P_{ij} \equiv \frac{f_{ij}}{\sum_j f_{ij}}$$

Terms whose CDE is above a threshold is unlikely to be specific since such terms are evenly distributed in many domains.

To appreciate the fact that a high frequency term will be more important in a domain, the CDE is further weighted by the term frequency in the particular domain when deciding which terms are important DSW's. Currently, the weighting method mimics the conventional TF-IDF method [10] in information retrieval. In brief, a word with entropy *Hi* can be think of as a term that spreads in *2**Hi* domains on average. The equivalent number of domains a term could be found then can be equated to $Nd_i = 2^{H_i}$. The term weight for w_i in the *j-th* domain can then be estimated as:

$$W_{ij} = n_{ij} \times \log_2 \left(\frac{N}{Nd_i} \right)$$

where N is the total number of domains. Unlike the conventional TF-IDF method, however, the *expected number of domains* that a term could be found is estimated by considering its probabilistic distribution across all domains, instead of simple counting.

The directory tree, after domain independent terms are removed from the associated documents, now represents a hierarchical classification of the domain-specific terms of different domains. The lists of domain specific words in the subdomains can thus be used for building domain-specific class models for disambiguating ambiguous words in various contexts.

The Appendix shows a list of highly associated domain-specific words of low cross-domain entropies and high term weights (with literal English translation) in 4 special domains [7].

6 DSW-Based Document Classification Model

Given the DSW's for various domains, some discriminative classification models have to be developed in order to make the best use of such information source. This is particularly true for DSW's that are sense ambiguous since sense ambiguity will make a DSW less useful as a representative of a domain unless quantitative measures related to the domain, such as the posterior probabilities of the term under various domains, are known.

For instance, if we use the DSW's simply as the (sense-insensitive) index terms in conventional VSM-based (Vector Space Model-based) search engines, and use them to calculate the similarity between documents for clustering without distinguishing the distinct senses of the same term, then we may classify a document into a class that has "relevant documents" including the same index term, such as the keyword "bank", but not the "interested documents" relevant to "the organization where money is saved or withdrawn".

This ineffective use of the index terms (or DSW's) results from the fact that a VSM does not take domain specificity (such as "the probability of the index term in a domain") into consideration when calculating the similarity between the query keywords and the document. Such ineffective use is also a main reason why people are calling for document re-classification or web mining at the backend of those quick (but dirty) search engines.

Inspired partly by such an observation, a Bayesian classifier, which incorporates the domain specificity measure as a posterior probability into the model parameters, is adopted.

With a Bayesian classifier, the document classification task can be formulated as finding the most probable domain label of a document, given the set of words w_1^n in a document:

$$\begin{aligned} d^* &= \arg\max_d P\left(d \mid w_1^n\right) \\ &= \arg\max_d P\left(w_1^n \mid d\right) P\left(d\right) \end{aligned}$$

d : class label of document
(i.e., the "domain")
w_1^n : words in the document

If we assume that a document is representable by the domain specific words in the document, we can simply restrict w_1^n to the set of DSW's of all domains. If we further assume that each domain specific word is generated independent of others, then we can simply have:

$$d^* = \arg\max_d \prod_{w_i \in DSW} P\left(w_i \mid d\right) P\left(d\right)$$

where $P\left(d\right)$ is the prior probability that the domain d is addressed, which can be estimated by the number of documents in d normalized by the number of all document. Furthermore, $P\left(w_i \mid d\right)$ represents the posterior probability that a DSW will appear in the domain d. Intuitively, the more a DSW in a document matches the right domain, the higher score one will have for that domain. It is this probability factor that provides the domain specificity

information. With this factor included, one can retrieve "interested documents" better, since an interested document relevant to "money" will now be retrieved not merely because it has the ambiguous keyword "bank" but other associated DSW's.

The above Naïve Bayesian assumption, however, has the problem on how to smooth the empirically obtained probabilities if some domain specific words do not appear in some domains. To make this smoothing issue well resolved, a maximum entropy-based classifier [1, 11, 8], is, instead, adopted so that word uni-gram counts in each domain can be used as a feature to estimate the posterior probability of the document, subject to the constrains that the expected feature values will fit the empirical counts, and all other unseen features are smoothed equally probable.

Given such a classifier, the text version of the web documents can be filtered with domain specific words so that only domain specific words in the training documents are used for training the class models (i.e., the posterior probabilities for each domain). The testing documents, filtered with domain specific words, are then classified based on the class models.

7 Classification Performance with Domain Specific Words

To see how the DSW-based text classifier achieves high classification accuracy with very few DSW's, a large collection of Chinese web pages was collected from a local news site. The HTML web pages are about 200M bytes pre-classified into 138 proprietary hierarchical domains of the news site (including the most specific domains at the leaf nodes as well as their parent domains towards the root). About 16,000 unique words are identified after word segmentation is applied to the text parts.

Totally, there are 4,279 documents in the collection. On average, each domain has about 30 documents. For simplicity, a held out estimate of the classifier performance, based solely on the DSW's, is adopted. (An n-fold cross validation would be better though.) The training set for the classifier consists of about 9/10 of the documents in each domain; the other 1/10 of the documents is used as the test set. Since some domains have only a few documents, they are excluded from performance evaluation to factor out biases introduced by insufficient evaluation data. The final set of documents for training and testing consists of 3,322 documents in 63 domains. Each domain has at least 23 documents for training.

To train the classifier models, the training documents are first word segmented with a word-unigram based word segmentation model [12, 9]. Those words that are not identified as domain specific words are then filtered out from the documents, leaving a bag of domain-specific words, like the one shown in Table 1, for model training. Each row in this table is derived from a sentence (or phrase) of the un-filtered document.

The set of domain specific words depends on the threshold applied to the list of word types in each domain, sorted by decreasing term weight Wij. The simplified document representation as shown in Table 1 is filtered by the top 5% words that have the highest term weights (hereafter, "the top 5% domain specific words" for short) of each domain. Although the full text is not shown, it is not difficult for a person to classify it into the "US-stock" domain if one is given this domain as a candidate. This characterization by DSW's is exactly the basis for document classification without resorting to the full text.

理財　股市　美國股市　企業　裁員　元月
個股　查詢
理財　股市　美國股市
企業　裁員　元月
紐約　五日
公司
二月　大幅
公司　經濟
裁減　的工作
二月　裁減　百分之
公司
消費　者的
企業　更多
趨勢
公司
裁員　第三
裁員
福特　汽車　裁員　汽車　業的　裁員
單月
汽車　裁員　產業
美國　勞工　市場
企業　裁員
日報

Table 1. A Document about "US-stock" filtered with the top 5% DSW's.
(Each row is a filtered sentence.)

The evaluation is based on the bags of words filtered with the top 20%, 10%, 5% and 2% of the domain specific words, each sampling step uses approximately half of the entries of the next higher sampling threshold.

Furthermore, since the TF-IDF (term frequency inverse document frequency) approach [13, 10] is widely used in information retrieval applications for indexing important terms within documents, it can also be applied to identify domain specific words in various domains. To make a comparison, the TF-IDF term weighting method is also applied to the same corpus to see the differences.

The training set performance and the test set performance are depicted in Figures 1 and 2, respectively, showing the percentages of accurately classified documents by using different sets of DSW's of different sampling thresholds. Three term weighting strategies are shown in the three curves, where the "CDE" model refers to the proposed term weighting method based on the cross-domain entropy measure, "IDF" refers to the traditional TF-IDF approach, and "RAW" refers to the case where all the words in the unfiltered documents are used for

training the classifier. The details of the two figures are also shown in Tables 2 and 3 respectively.

Note that, the RAW model is equivalent to using the constant threshold of 100% for extracting domain specific words from the word list sorted by decreasing term weights. The performance is therefore a constant, shown as a horizontal line, in comparison with other models with different thresholds.

It is obvious from the curves that the performances of the other two models will eventually drop to the horizontal line if one tries to increase the number of DSW's to achieve "better" performance. This is not surprising since the model will then be contaminated by other non-specific words. And this is exactly what the domain specific words are valuable. *The performance curves at the low sampling thresholds are therefore the major criteria for comparing various models* [13].

In this area with low sampling thresholds, the CDE model is consistently better than the traditional TF-IDF approach, even though the differences are small. And, it breaks the performance of the RAW model much faster.

One must not be over-optimistic to the test set performance shown here, though, since the number of documents under testing is only 301 documents. With 5% of most heavily weighted terms, there are only 3 mis-classified documents. Hence it quickly reaches the highest performance when the domain specific words are doubled to 10%. Due to the small number of test documents, the performance figures may have a great variance. A more conservative performance might be acquired if the performance is evaluated against a larger data set. However, as a conservative estimate, it would be safe to say that with 5~10% most heavily weighted terms, a 99% accuracy is possible for this task.

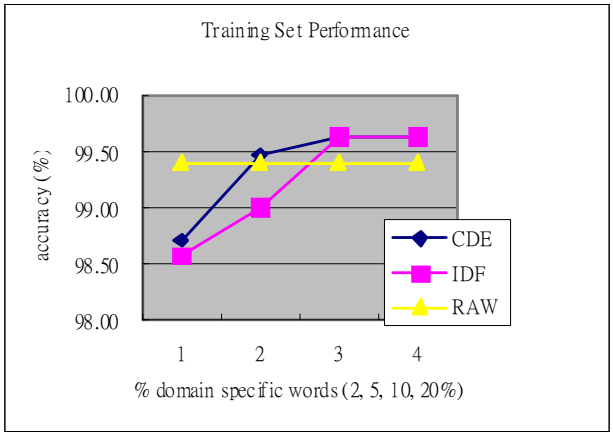

Figure 1. Training Set Performance

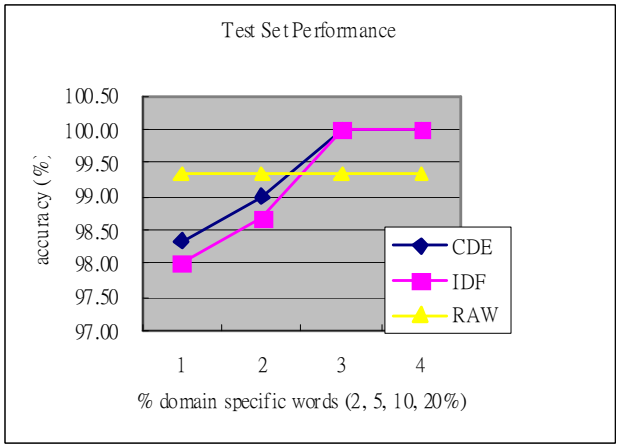

Figure 2. Test Set Performance

%DSW	CDE	IDF	RAW
2	98.71	98.58	99.40
5	99.47	99.01	99.40
10	99.64	99.64	99.40
20	99.64	99.64	99.40

Table 2. Training Set Performances

%DSW	CDE	IDF	RAW
2	98.34	98.01	99.34
5	99.00	98.67	99.34
10	100.00	100.00	99.34
20	100.00	100.00	99.34

Table 3. Test Set Performances

8 Domain Specific Words Acquisition Results

Although the DSW acquisition performance is not the main focus of this paper, it is interesting to summarize some of the results in [7].

In 5 sample domains of small sizes (with 300 word types or less), the current approach extracts domain specific words at an average precision of 65.4% and an average recall of 36.3%, corresponding to 45.8% F-measure, if the top-10% words with highest term weights are extracted as domain-specific words. In other words, by only gathering the first 10% of the whole word list, the current term weighting measure can identify about 36% of the embedded domain specific words, and one can identify significant amount of DSW's about every 1.5 entries from the top-10% list of low entropy words.

In comparison with the popular TF-IDF (term frequency inverse document frequency) term weighting approach, it is observed that the top-20% sampling threshold for the baseball domain results in 51.6% F-measure with the CDE-based approach, as opposed to 47.8 % with the TF-IDF weighting method.

From all the observations, the CDE measure does indicate the "degree of specificity" more informatively. In particular, the degree of domain specificity of a term is estimated by considering the cross-domain *probability distribution* of the term in the current CDE-based approach. In contrast, the TF-IDF approach simply counts the number of domains a term is found as a measure of randomness. The CDE approach is therefore gaining a little bit performance than a TF-IDF model.

The results partially confirm that one can extract a large domain specific lexicon at little cost from the web pages by removing non-specific words from web documents.

9 Concluding Remarks

In this paper, a method for learning domain specific class models for nodes of a hierarchical web document tree from domain specific words associated with each node is proposed for web document classification. With this approach, web documents can be easily re-classified into an existing web hierarchy directly, instead of being re-clustered with other documents to form a hierarchy that is unlikely to fit any human classification criteria.

With only the 5~10% of most heavily weighted DSW's and a maximum entropy based classifier, classification accuracy of about 99% is observed when the method is evaluated on a task that classifies web documents into the 63 domains of an existing news web site. Furthermore, the performance of the current term weighting method is consistently better than the conventional TF-IDF approach in the current task, in terms of classification performance and domain specific word acquisition performance. The current approach is therefore an appropriate candidate for applications of this kind.

Appendix: Sample Domain Specific Words

Baseball	Broadcast-TV	Basketball	Car
日本職棒 (Japanese professional baseball)	有線電視 (cable TV)	一分 (one minute)	千西西 (Kilo-c.c.)
棒球賽 (baseball games)	東風 (Dong Fong TV Station)	三秒 (three seconds)	小型車 (small car)
熱身 (warm up)	開工 (start to work)	女子組 (girl's teams)	中古 (used car)
運動 (athlete)	節目中 (on air)	包夾 (fold; clip)	引擎蓋 (engine cover)
場次 (time table)	廣電處 (radio-tv office)	外線	水箱 (tank)
價碼 (cost)	收視	犯規 (foul)	加裝
球團 (baseball team)	和信 (Ho-Hsin TV Station)	投籃 (shot)	市場買氣 (market atmosphere)
部長 (manager)	新聞局 (government information office)	男子組 (male team)	目的地 (destination)
練球 (practicing)	開獎	防守 (defense)	交車 (car delivery)
興農 (Hsin-Lung team)	頻道 (channel)	冠軍戰 (championship)	同級 (of the same grade)
球場(course; diamond)	電視 (TV)	後衛 (fullback)	合作開發 (co-development)
投手 (pitcher)	電影(movie)	活塞 (Piston team)	安全系統 (safety system)
球季 (season)	熱門 (hot)	國男 (national male team)	行李 (luggage)
賽程 (schedule)	影視 (video)	華勒(Wallace)	行李廂 (trunk)
太陽 (the Sun team)	娛樂 (entertainment)	費城 (Philadelphia)	西西 (c.c.)

Appendix. Sample domain specific words with low cross-domain entropies in 4 special domains [7].

References

[1] Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra, "A Maximum Entropy Approach to Natural Language Processing." *Computational Linguistics*. 22(1):39-71, 1996.

[2] Dmitry Davidov, Evgeniy Gabrilovich, Shaul Markovitch, Parameterized generation of labeled datasets for text categorization based on a hierarchical directory. *Proc. of SIGIR 2004*, pp. 250-257, 2004.

[3] Dumais & Chen, "Hierarchical Classification of Web Content", SIGIR-2000, 2000.

[4] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey, "Scatter/ Gather: a cluster-based approach to browsing large document collections." In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 318--329, June, 1992.

[5] Glover, Eric J., Kostas Tsioutsiouliklis, Steve Lawrence, David M. Pennock, and Gary W. Flake, "Using web structure for classifying and describing web pages." In *Proc. 11th WWW Conference*, pages 562—569, 2002.

[6] Huang, C.-C., S.-L. Chuang, L.-F. Chien, "LiveClassifier: Creating Hierarchical Text Classifiers through Web Corpora", pp. 184-192, WWW 2004, 2004.

[7] Jing-Shin Chang, "Domain Specific Word Extraction from Hierarchical Web Documents: A First Step Toward Building Lexicon Trees from Web Corpora," *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing,* pp. 64-71, International Joint Conference on Natural Language Processing (IJCNLP-05), Jeju Island, Korea, 2005.

[8] Laird Breyer, "The DBACL Text Classifier," http://www.lbreyer.com/gpl.html, 2004.

[9] Ming-Yu Lin, Tung-Hui Chiang and Keh-Yih Su, "A Preliminary Study on Unknown Word Problem in Chinese Word Segmentation," *Proceedings of ROCLING VI*, pp. 119-142, 1993.

[10] Ricardo Baeza-Yates and Berthier Ribeiro-Neto, *Modern Information Retrieval,* Addison Wesley, New York. 1999.

[11] Ronald Rosenfeld, "A Maximum Entropy Approach to Adaptive Statistical Language Modeling," *Computer, Speech, and Language,* 10(3):187-228, 1996.

[12] Tung-Hui Chiang, Jing-Shin Chang, Ming-Yu Lin and Keh-Yih Su, "Statistical Models for Word Segmentation and Unknown Word Resolution," *Proceedings of ROCLING-V*, pp. 123-146, Taipei, Taiwan, R.O.C. 1992.

[13] Yang, Yiming and Jan O. Pedersen, "A comparative study on feature selection in text categorization." In *Proceedings of the International Conference on Machine Learning*, pages 412—420, 1997.